趣味の技術入門

●よくわかる

マイコン入門

山科敦之
吉田紀彦

日本図書館協
選定図書

新星出版社

赤い星は……知識の星
実用図書の新星出版社

● やさしい電子オモチャの作り方 ★明甎泰平著
● やさしいラジコン飛行機入門 ★畠山　泉著
● やさしいラジコン自動車入門 ★水谷則雄著

よくわかる
マイコン入門

山科敦之★吉田紀彦著

はじめに

機械というものは、たいてい融通のきかないでくのぼうです。自動車にしても、ハンドルを右にきれば何が何でも自動車は右にまがり、ハンドルを左にきれば何が何でも自動車は左にまがります。たとえ、進行方向に壁があっても、人がいても、犬や猫がいても、自動車は自分で勝手によけたりはしてくれません。自動車を運転する人間が間違えれば、間違えたように自動車は動きます。機械は、「考える」能力がないからです。

ところが、「考える」機械もあります。それは、コンピュータと呼ばれる機械です。たしかに、まだ完全ではありませんが、それでもほかの機械と違って、ある程度、人間のかわりに考えてくれる能力を持たせることができます。

例えば、数の計算はもちろん、チェスの相手をさせたり、多くの資料の中から欲しい資料を取り出させたり、郵便番号を読み取って郵便物を分類したり、今まで人間にしかできなかったことをしてくれるようになっています。

コンピュータは、十年くらいまえなら、大きくて高価なもので、とても一人で買えるようなものではありませんでした。大きな会社や国の役所のような

ころが、何億というお金を出して買うようなものでした。

それが、ICというもののおかげで、小さくて、安いコンピュータが作れるようになってから、趣味としてのコンピュータが売られるようになってきました。最近よく耳にする、マイクロ・コンピュータ、略してマイコンが、それです。

この本では、人工知能とも呼ばれるコンピュータの小型判、マイコンの大体の仕組みについて、みなさんが理解できるようやさしく書いたつもりです。コンピュータの基本的な動きや、コンピュータの取り扱う情報などについても、なるべくやさしく書きました。この本が、マイコンを理解し活用するための足がかりとなれば、と願っています。

この本を書くにあたって、稲野雅彦氏、堀健一氏、ヌル・ソクラテス氏の御協力をいただきました。ほかに、こころよく写真を貸して下さった東芝、日本電気、パナファコムの方々にもお礼を申し上げます。

山科敦之
吉田紀彦

目次

第1章　マイコンの誕生……9

マイコンの生い立ち……10

ICの登場……14

第2章　マイコンの仕組み……19

2進法って何だろう……20

ビットとは何だろう……26

マイコンの基本的機能……35

マイコンには五つの働きがある……35

マイコンの構造……42

CPUとは……46

演算論理回路……47

制御回路……58
レジスタ部……59
メモリ部……60
アドレス線とデータ線……61
ROM……62
RAM……67
入出力装置……69
I／Oポート……69
入出力装置の種類……70
マイコンの情報の流れ……74
ハードウェアとは……78

第3章　マイコンと話をする……79

人間の言葉とコンピュータの言葉……80
コンピュータの言葉……85
機械語……85

アセンブリ言語……89
高級言語……92
プログラムって何だろう……96
プログラムを作るには……98
フロー・チャート……99
プログラミング……105
デバッグ……134

第4章　マイコン・キットのいろいろ……137

DO　IT　YOURSELF……138
脚の多い部品を逆に取りつけてしまった時の取りはずし方……139
ヒゲやブリッジの除き方……141
電源を甘くみないこと……142
マイコン・キットのいろいろ……142
日立　H68／TR……143
東芝　EX—80……148

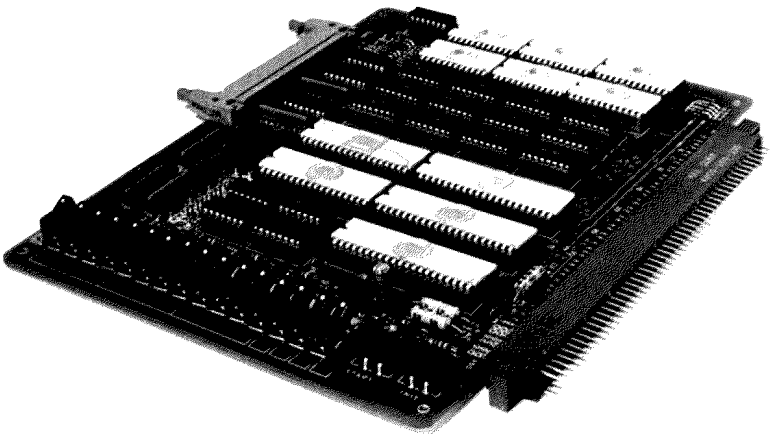
日本電気（NEC）　TK—80……152
パナファコムLKIT—16……154
そのほか……155

第5章　マイコンに何ができるか……157

人工知能……158
マイコンの応用……160
産業での応用……161
マイコンで遊ぶ……162

第6章　マイコン用語の解説……171

★第1章
マイコンの誕生



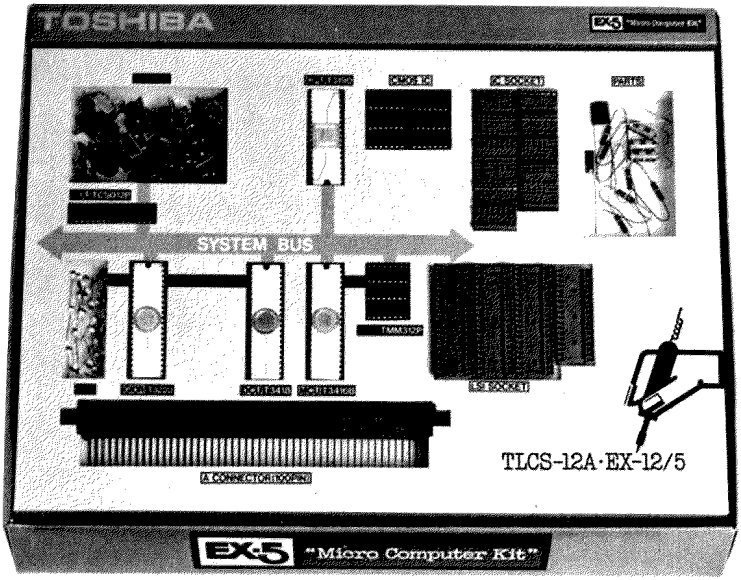
東芝・ＥＸ－５

マイコンの生い立ち

あの大きな部屋を占めていたコンピュータが、手のひらに乗るくらいに小さくなったのは、わずか、三〇年たらずなのです。

最初は、アメリカのハーバード大学で、ＡＳＣＣ―ＭＡＲＫ・Ｉが一九四二年に作られました。これは、試験的に作られたもので、歯車などを使った実用性の薄いものです。続いて、ペンシルバニア大学で、ＥＮＩＡＣが一九四六年に作られました。これは真空管を使ったもので、一五〇キロワットもの電力を必要としたといわれます。

ノイマンという学者の意見に従って、さらに進んだ形のもので、基本的には現在のコンピュータとほぼ同じものが作られました。ケ



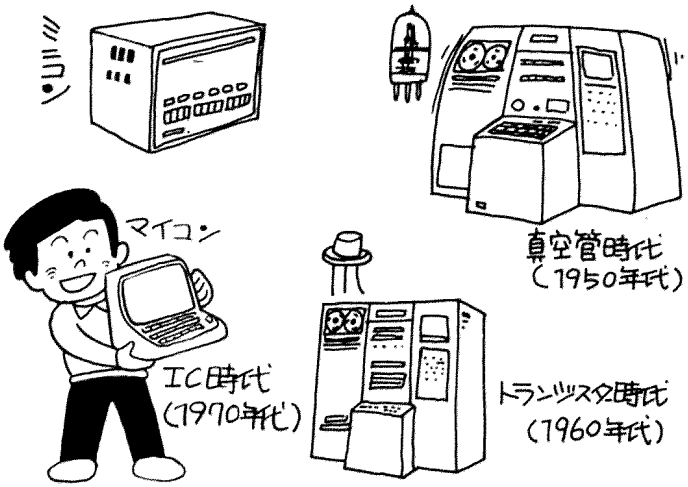
パッケージされたマイコンキット

ンブリッジ大学のEDSAC（一九四九年）とペンシルバニア大学のEDVAC（一九五二年）です。EDVACが世界中に三台もあれば、あらゆる計算問題がこなせる、とある学者がいったというエピソードも残っていますが、現在のマイクロ・コンピュータの方が、やはり、それより優れています。

アメリカで、コンピュータが商品として売り出されたのは、一九五二年からです。日本では、しばらくの間、試作時代が続きます。

一九五二年から一九五七年までを、コンピュータの歴史では、第一世代、あるいは真空管の時代と呼んでいます。この頃のコンピュータは真空管を使っているからです。

第一世代のコンピュータは、真空管を何万本も使った、非常に大きなものでした。発生する熱量も多いうえに、真空管は、比較的寿



コンピュータの歴史

命が短いので故障が多く、不安定な面がありました。

そこで、真空管にかわって、トランジスタが登場します。

一九五八年から一九六三年頃までは、第二世代、あるいはトランジスタの時代と呼ばれています。

トランジスタは真空管に比べて小さく、寿命も半永久的なので、第二世代のコンピュータは、第一世代のコンピュータより、小さく、安定したものになったのです。

第二世代の頃には、プログラミング・システムが開発されました。プログラミング・システムというのは、プログラム用の言葉の体系のことです。

それまで、人間がコンピュータに命令を与える場合、数字だけで構成された機械語を使

っていました。

数字ばかりが並んでいる機械語は、普通の人間に理解できないのはもちろん、専門家にとっても扱いにくいものだったのです。かといって、コンピュータに人間の言葉を教えこむことは、不可能に近いことです。

なぜなら、コンピュータは、ごく単純なことを、短時間に、しかも正確に行うことができますが、人間の言葉のように複雑で例外が多いものは理解できないからです。

そこで、人間の言葉と、機械語との中間に位置するプログラム用言語が作られたわけです。プログラム用言語は、単純な文法と、限られた単語とで成り立っているもので、これなら人間も比較的覚えやすく、コンピュータにも教えやすいものです。

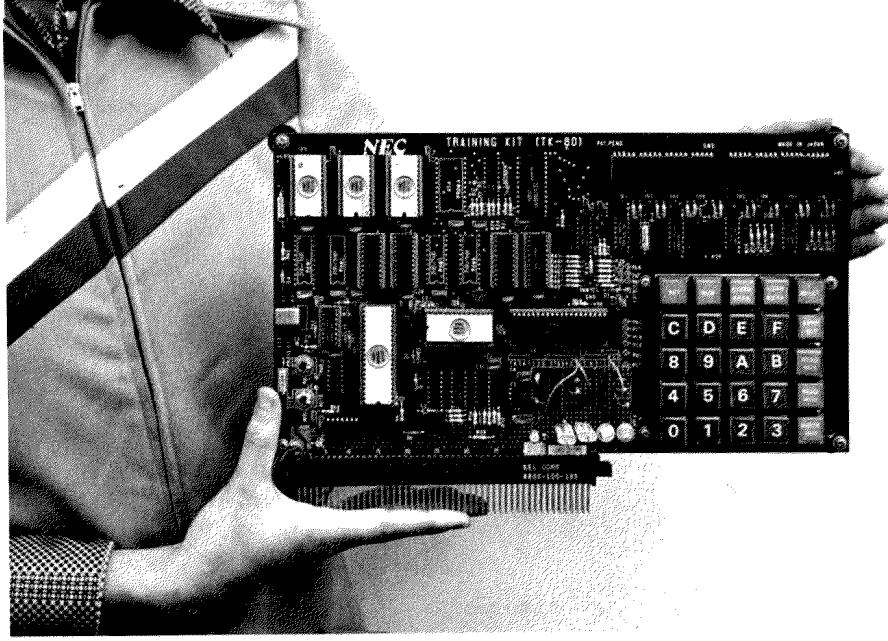
ただし直接その言葉をコンピュータが理解するのではなくて、コンピュータの中に翻訳器が内蔵されていて、それを仲介にしてコンピュータはプログラムを理解するのです。

プログラム用言語には、アセンブリ、ベーシック、フォートラン、コボルなどがあります。

また、同じ頃に、現在のオンライン・システムなどが開発されました。これは、中央の大型コンピュータと各地の端末装置とが直接情報交換のできるシステムです。これによって、情報の流れが速く確実になり、また多量の情報を処理することが可能になりました。

現在は情報の時代である、とさかんにいわれます。情報を管理するのはほとんど、コンピュータの仕事になってきています。

マイコンキットは、もともと企業が、コンピュータを導入する時に、コンピュータの基



NEC・TK－80

本的な仕組み、機能を勉強するために作られたものです。けれども一般の人にも購入する人が増えてきて、まるでオモチャのように簡単に取り扱うことができるようになりました。

ICの登場

マイクロ・コンピュータができたのは、IC（集積回路）のおかげです。

トランジスタが小さいといっても、トランジスタを何万個も組入れたコンピュータは、やはり相当大きなものになります。ところが、ICは、トランジスタ百個から数万個分の働きを持った、数ミリ四方のものなので、ICを使ったコンピュータは、タイプライターほどの大きさにまで縮めることができるようになりました。洗濯機や電子レンジ用なら、キ

ャラメル程度の大きさに縮めて、それぞれの機械の中に組入れることもできます。しかも、大量生産ができるので、値段も安く、マイクロ・コンピュータを組入れても、たいして価格は高くなりません。

ICとは、数ミリ四方のシリコン・チップ（ケイ素の小片）に、百個から数万個分のトランジスタ回路を「印刷」したものです。「印刷」といっても、本のようにインクで印刷するのではありません。ただ、製造工程が印刷とよく似ているのです。

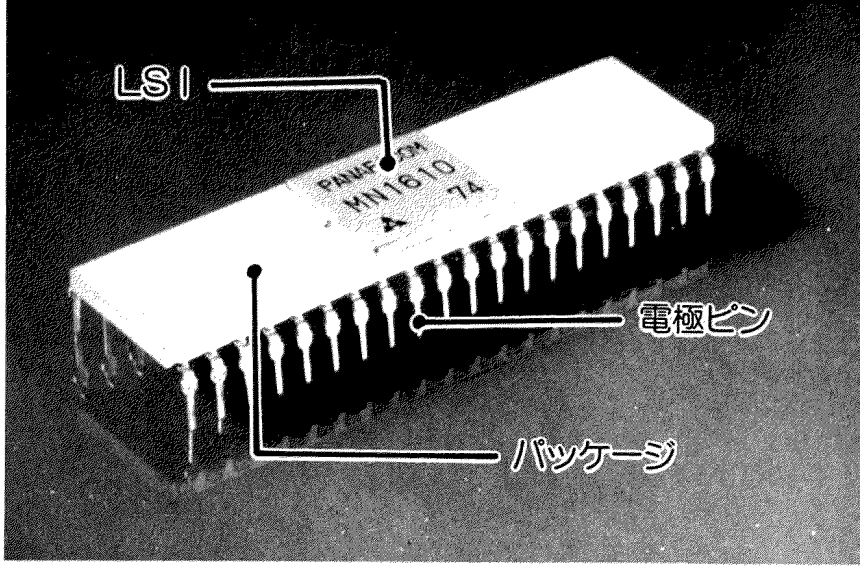
大きく書いたトランジスタ回路を写真ネガで撮って、それを表面加工したシリコン・チップに焼きつけて、現像・定着などの加工を施して、アルミニウムを同じように取り付ければ、ICのできあがりです。

ICには、大別して四つの種類があります。

○SSI（小規模集積回路）
　トランジスタ百個分以下
○MSI（中規模集積回路）
　トランジスタ百個分から千個分
○LSI（大規模集積回路）
　トランジスタ千個分から数万個分
○超LSI
　トランジスタ数万個分以上（ただし、これはまだ研究の段階です）

SSI、MSIを二百から三百個使ったコンピュータを、ミニコンといいます。これは、価格が何百万もするもので、大型コンピュータに近い働きをします。

これに対して、マイクロ・コンピュータ、すなわちマイコンは、LSIを一個から数個使ったもので、価格もマイコン・キットだと十万円前後になります。機能にしても、いろ



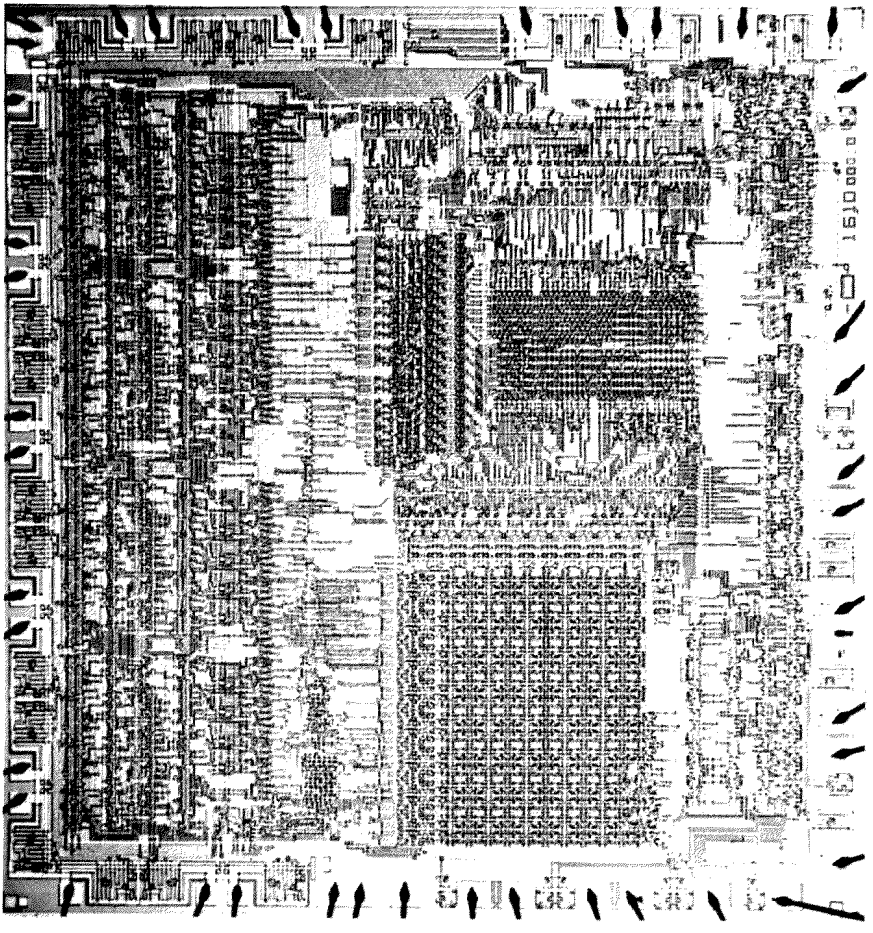
マイクロ・プロセッサの外観（パナファコム・MN1610）

いろな周辺装置（テレタイプやテレビ受像機など）を取り付ければ、活用できる範囲は非常に大きいものです。

マイコンといってもいろいろあります。ミニコンにかわるものとしてのマイコンもありますし、趣味としてのマイコン・キットなどのほか、産業用として、工場機械を制御するものや、家庭電気製品に組込まれるもの、ゲーム機械に組込まれるもの、自動販売機に組込まれるもの、カメラやスーパーマーケットのレジなどに組込まれるもの、などがあります。

マイコンの主要部分となるＬＳＩを、特にマイクロ・プロセッサといいます。

マイコン・キットでは、マイクロ・プロセッサのほかに、いくつかのＬＳＩを接続しなければなりませんが、まえにあげた機械に組



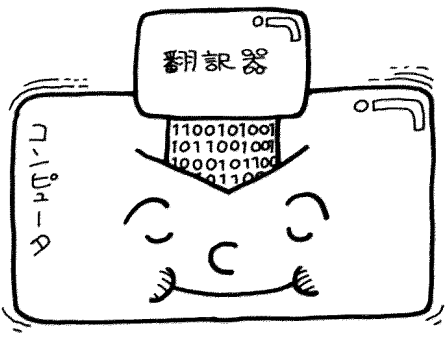
マイクロ・プロセッサの中心部分・LSIの内部拡大写真

（パナファコム・MN1610）

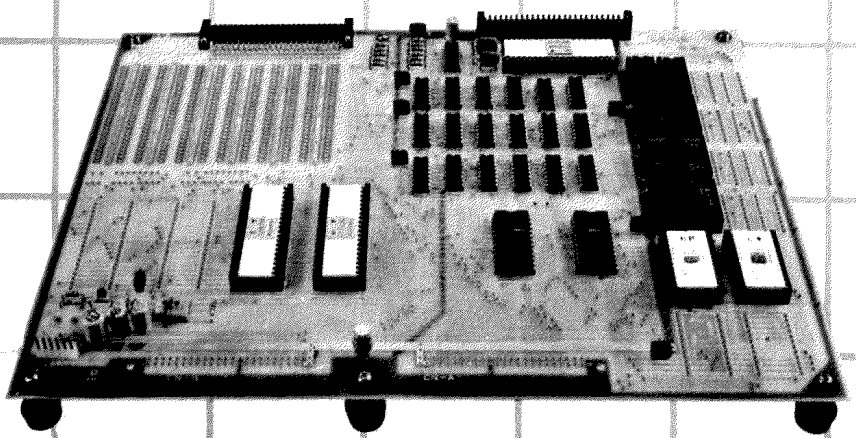
込まれるマイコンには、たった一つのＬＳＩに、それらの全てを組込んだものがあります。これを、たった一枚の小片でできたコンピュータという意味で、ワンチップ・マイクロ・コンピュータと呼ぶことがあります。マイコン・キットは、一枚の板の上に作られているので、ワン・ボード・コンピュータといわれます。

世界で最初に発表されたマイクロ・プロセッサは、インテル社と日本の電卓メーカーとで共同開発された、ｉ四〇〇四というＬＳＩです。それは一九七〇年代に入って、すぐのことです。続いてｉ八〇〇八というマイクロ・プロセッサも発表されました。それ以来、毎年のように各社からマイクロ・プロセッサが発表されて、現在に至っています。

それでは、マイクロ・プロセッサと、ほかのＬＳＩとが、どのように働きあって、マイコンを形造っているか、第2章で説明しましょう。



★第2章
マイコンの仕組み



マイコンは、大きくわけて三つの部分からできています。

もっとも重要な働きをするCPU、記憶する働きをもつメモリ部、人間でいえば目や耳、口や手足にあたる入出力装置の三つです。

この三つの部分が、おたがいに連絡しあって、いろいろな仕事をするのですが、それを説明するまえに、2進法とビットについて説明しておきます。なぜなら2進法とビットを知っておくと、これから述べるマイコンの仕組みの説明が理解しやすいからです。

2進法って何だろう

私達が普段使っている数は10進法です。1から数えて行って、10ごとに一つ桁上がりで書くので10進法というのですが、普段、何げなく使っているので、私達は10を一つのまとまりと感じるようになっています。けれども、自然界をよく観察すると、決して10は特別な数ではありません。木に繁る葉は、たいてい三五二四枚とか、七一三九枚とか、私達にとっては、はんぱに感じられる枚数の場合が多く、かえって、二〇〇〇枚ちょうどだったりすると、「珍らしく」思われます。10を一つのまとまりとして扱っているのは、お札の金額とか、何かの目盛の区切りとか、全て人間が作ったものに限られています。

実は、10進法は数を書き表わしたりする時の、勝手な約束に過ぎないのです。一説には、人間の指が10本だったので、10進法が使われるようになったのだといわれます。火星人の指が全部で七本だとすると、多分火星人は七進法を使っているでしょう。

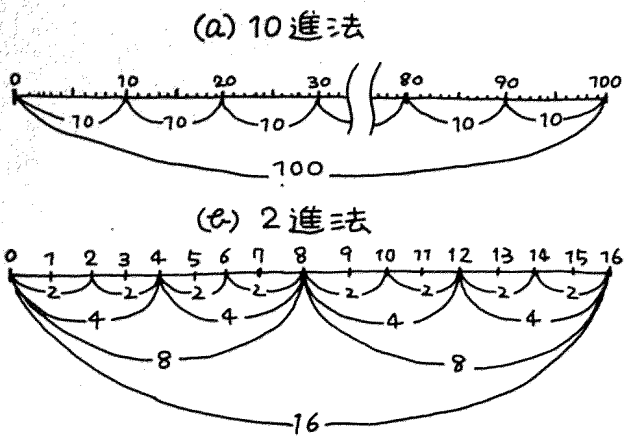


図2－1　2進法と10進法

数は、実際のところ、どこまでも続く、区切りのない直線のようなものです。何の区切りも、初めは存在していないのです。

そこで、私達が知らず知らずのうちに設定している「10」という目印を忘れます。

さて、数に区切りはなくなりました。が、やはり何の目印もないと不便です。

今度は、あらためて二ごとに目印をつけてみます。その次に、二を二つずつまとめて目印をつけます。これは、一〇を一〇ずつまとめて一〇〇とするのと似たようなことです。二を二つずつまとめたもの（つまり四）を、さらに二つずつまとめて目印をつけます。これは、一〇を一〇ずつまとめたもの（つまり一〇〇）を、一〇ずつまとめて一〇〇〇とするのによく似ています。以下同じようにして行けば、10進法の、一〇、一〇〇、一〇〇〇、

などと同じように、二、四、八、一六、という目印がついたことになります。（図2―1）

この目印を用いて、数を表わすやり方を2進法といいます。同じことが5進法にも、8進法にも、16進法にもいえます。

10進法は10ごとに桁あがりするので、一、二、三、四、五、六、七、八、九まで数えて「じゅう」のところで、一〇と二桁で書きます。同じように、九九の次に桁あがりして、一〇〇となります。

それとまったく同じことを2進法でも行います。2進法は、2ごとに桁あがりするので、一と数えて、「に」のところで一〇と桁あがりするのです。同じように、一一の次は一〇〇と桁あがりします。

2進法での一〇や一〇〇を「じゅう」「ひゃく」と読んではまぎらわしいので、小数を読むときのように、一〇は「いちぜろ」一〇〇は「いちぜろぜろ」と読むことになっています。これはそれぞれ二、四のことです。

また、単に10とか100とか書いても、それが10進法なのか2進法なのかわからないので、10進法の時はそのままですが、2進法の時は、$(10)_2$とか$(100)_2$というように、2進法であることを示すことがあります。あるいは、$10_{(2)}$、$100_{(2)}$と書くこともあります。もちろん、特に示す必要のない時は、そのままで何にもつけません。

指数を使って、もっと簡単に説明しましょう。例えば293という数は、

$$\dot{2}\dot{9}\dot{3}=\dot{2}\times10^2+\dot{9}\times10+\dot{3}\cdots\cdots\cdots\cdots(イ)$$

と書くことができます。百の位の2は、$200=2\times10^2$、十の位の9は$90=9\times10$を表わしていることが、この式ではっきりとわかりま

す。そこで、10や10^2のかわりに2や2^2などを使うと、

$293=1\times2^8+0\times2^7+0\times2^6+1\times2^5+0\times2^4$
$+0\times2^3+1\times2^2+0\times2+1$……(ロ)

という式になります。この式の掛け算の部分の頭だけをとりあげてならべてみると、

100100101

となりますが、この数字は293を2進法で表わしたものなのです。このことから、2進法というのは、(イ)の式のように書かれた数を、(ロ)式のように書き直すことだともいえるでしょう。

2進法で書かれた数を10進法に直すのは簡単です。例えば、1011, 1100010, (どちらも2進法)の二つの数を10進法に直してみましょう。表2—1を参照して下さい。

$(1011)_2=1\times2^3+0\times2^2+1\times2+1$

2進法	10進法
0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7
1000	8
1001	9
1010	10
1100000	96
1100001	97
1100010	98
1100011	99
1100100	100
1100101	101
使用する数字	使用する数字
0.1	0. 1. 2. 3. 4. 5. 6. 7. 8. 9

表2－1　10進数と2進数の比較

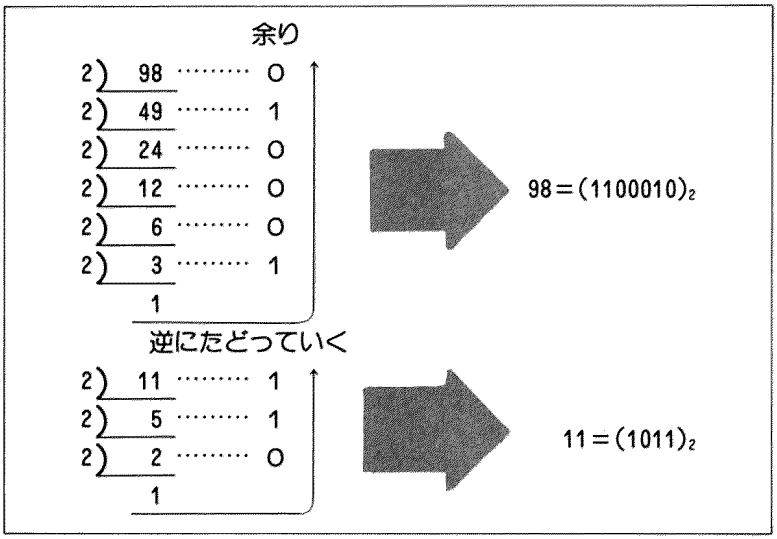


図２－２　10進数を２進数に直す方法

$$=1\times2\times2\times2+0\times2\times2+1\times2+1$$
$$=1\times8+0+1\times2+1=8+2+1=11$$

$$(1100010)_2=1\times2^6+1\times2^5+0\times2^4+0\times2^3+0\times2^2+1\times2+0$$
$$=1\times64+1\times32+0\times16+0\times8+0\times4+1\times2+0$$
$$=64+32+2=98$$

数が大きくなると計算が面倒にはなりますが、すべての２進数は、このようにして10進数に直すことができます。

10進法で書かれた数を２進法に直す時も、考え方は今行った方法の逆になるわけですが、実際には計算がとても面倒になります。それで、図２―２のような方法で計算するのが普通です。

10進法の数を次々に２で割って行って、その時の余りの数を右に書いておきます。最後に商が１になったところでやめ、その１から余りを逆にたどって行けばいいわけです。

2進法を使って計算をする時は、10進法で計算する時に比べて、非常に単純になります。たとえば、掛け算の場合にしても、10進法では九九を覚えなければなりませんが、2進法では、「一一」とでもいえそうな単純で量の少ないものしかありません。（表2―2）

1を4回も5回も足さなければならない時は、1＋1＋1＋1＝(1＋1)＋(1＋1)＝10＋10＝100のように、二つずつに分けて計算すると考えやすいでしょう。数が2進法で書かれているだけで、基本的には10進法の時と変わりません。引き算、割り算もやはり10進法の時と同じようにしてできますが、こちらは慣れないととてもできません。それに、2進法で計算する人は、まずいません。桁数が10進法に比べて多いうえに、慣れていないからです。基本的な計算規則は単純ですが、実際に行ってみるとわかるように、途中で頭がくらくらしてきます。

ところが、コンピュータはこの2進法でしか計算できないのです。しかも、足し算しかできません。では、コンピュータに足し算以外

足し算	掛け算
1＋1＝10 1＋0＝1 0＋0＝0	1×1＝1 1×0＝0 0×0＝0
例 10110 ＋) 11011 110001	例 10110 ×) 11011 10110 10110 10110 10110 1001010010

表2－2　2進法の足し算と掛け算

の計算を行わせる時はどうするのか、不思議でしょう。その不思議な気持を残したままにしておいて下さい。後で説明します。

さて、2進法のことが大体わかったところで、「ビット」の説明をします。

ビットとは何だろう

ビット（bit）という言葉は、binary digit（バイナリー ディジット）を縮めてできた言葉です。binary digitは、「2進法の数字」という意味を持っています。英単語にbit（ビット）（少量、少し）という言葉がありますが、直接関係はありません。

ビットとは、情報量を計るための基本単位として使われます。ある情報がどのくらいの大きさを持っているかの目安となります。では、情報量とはどういうことでしょうか。それを考えるまえに、情報とは一体何なのか考えてみなければなりません。

情報とは、役に立つ報（しら）せのことです。何かある報せを受け取った人が、それを利用して考えたり、行動したりする場合、その報せを情報と呼ぶのです。

例えば、わかりきっていることを受け取った場合、その報せは情報とはいえません。その報せを受け取ることで何かの役に立つということがないからです。あいうえおなどは平仮名ですという報せを受けても、そんなことはたいていの人が知っているので、その人達にとっては情報ではないわけです。

また、英語を知らない人が、英語で報せを受けても、それは情報にはなりません。その報せを理解することができないのですから、やはり何の役にも立たない無意味な音でしか

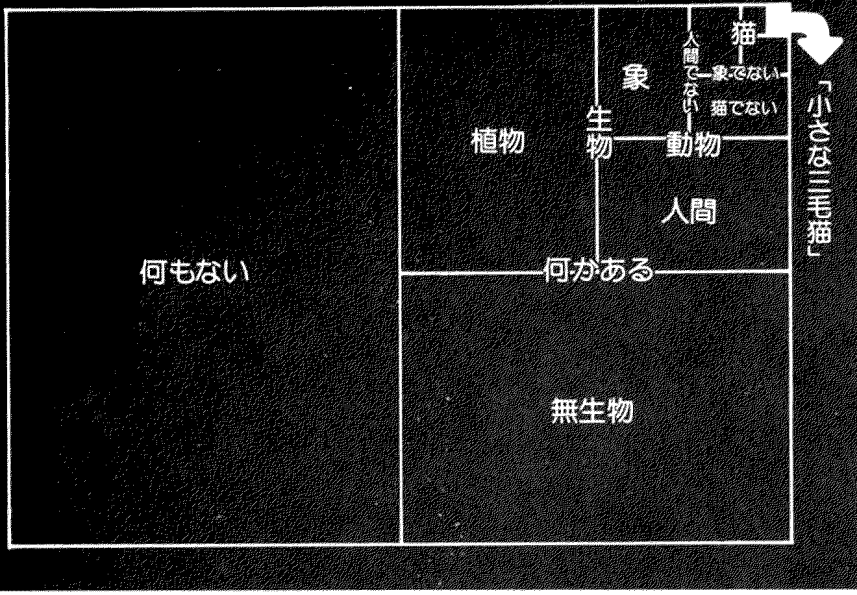


図２－３　壁の向うに何があるか（情報）

ないのです。

情報が伝えられるというのは、何かが知らされるということです。何かを知るということは、たくさんありうることのなかの、どれか一つであることがわかるということです。

あなたが今、壁のまえに立っているとします。壁の向うは見えません。それでは、壁の向うに何かがあるでしょうか。

あなたは、今、その質問に答えることができません。何も知らないからです。

壁の向う側の人が、

「何かあるぞ」

と教えてくれたとします。あなたにわかったことは、何かがあるということだけですが、少なくとも、

・何かが　ある
・何も　ない

の二つのうち、「何かがある」ことを知ったことになります。すなわち、情報を受けとったことになります。

何かがあるとすれば、「何が」あるのでしょうか。

まず考えられることは、無生物か生物かのどちらかです。

「生物だ」

と教えてもらえれば、次に考えられることは動物か植物かのどちらかです。動物だとわかれば、次に考えられるのは、人間かそうでないかということです。人間でなければ、象か象でないか、象でなければ、猫か猫でないか、猫であれば、大きいか小さいか、色は黒いか黒くないか、赤いか赤くないか、三毛か三毛でないかということになって行きます。

このようにして、あなたは壁の向うに、小さくて可愛らしい三毛猫が坐っていることを知るのです。情報とは、このように伝えられて行くものです。(図2—3)

壁の向うに、何かがあるかないかもわからない状態から、三毛猫がいると知るまで、その時、その時に考えられる状態を大きく二つに分けて、そのどちらであるかをいうことによって、より詳しいことを知らせて行く仕組みが情報のもっとも基本的な仕組みです。

「壁の向うに小さい三毛猫がいる」

という情報は、これ一つで、図2—3のような事柄を伝えることになります。

人間は言葉によって、実は複雑な事柄を、ごく単純に伝えてしまいますが、コンピュータにはそんなマネはできません。

人間は、ごちゃごちゃしたものごとをたくさんに区切って、それを一つ一つ区別するた

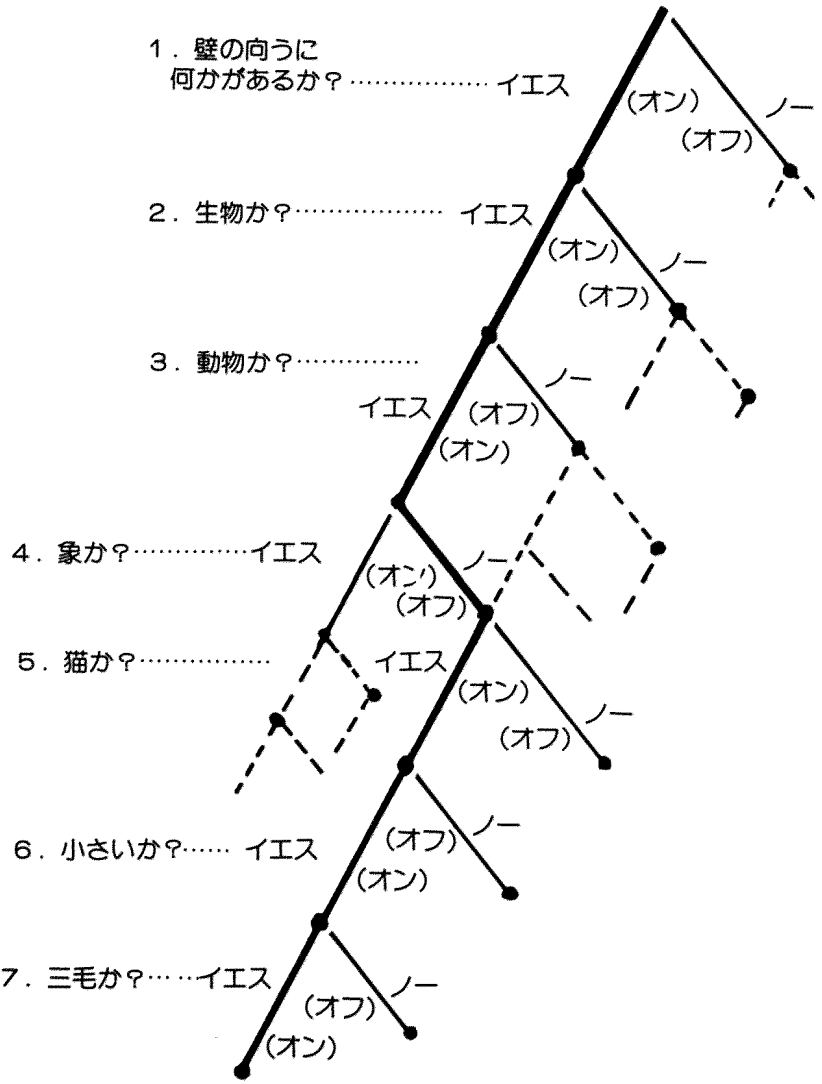


図２－４　壁の向うに何があるか

めに名前をつけています。例えば、人間の体は、はっきりとした境界のないものですが、それを人間はいくつかに分けて、頭とか、手足とか、人差指とかいって区別しています。そういう区別があるからこそ、情報は伝わるのです。

コンピュータの中では、電気信号に変換された情報がかけめぐっているのですが、コンピュータは情報を扱うものだから、どうしても、ものごとを区別する必要があるのです。

けれども、人間のような言葉を使うことはできません。コンピュータが使っているのは電気信号だけです。

電気には、オンとオフの二つの状態があります。コンピュータに区別できるのは、この二つだけです。電気信号があるか、ないか、つまり、オンかオフかの二つだけで、多くのものごとを区別しなければならないのです。

ところで、人間の言葉にもオン、オフと似たようなものがあります。イエスとノーです。すなわち、肯定と否定の二つです。

図2—3を、図2—4のように書きかえてみれば、情報の仕組みが多くのイエスとノーの組合わせで説明できることがわかるでしょう。

これをそのまま、オンとオフで置きかえれば、オンとオフを組合わせることで、小さな三毛猫が壁の向うにいるという情報を表わすことができるということになります。

オンを1、オフを0で表わすことにすれば情報は1と0との組合わせで表わすことができることになります。

図2—3、2—4は、具体的な質問に対する肯定（イエス）（オン＝1）と、否定（ノー）（オフ＝0）の組合わせの例でしたが、この質問をいろいろ

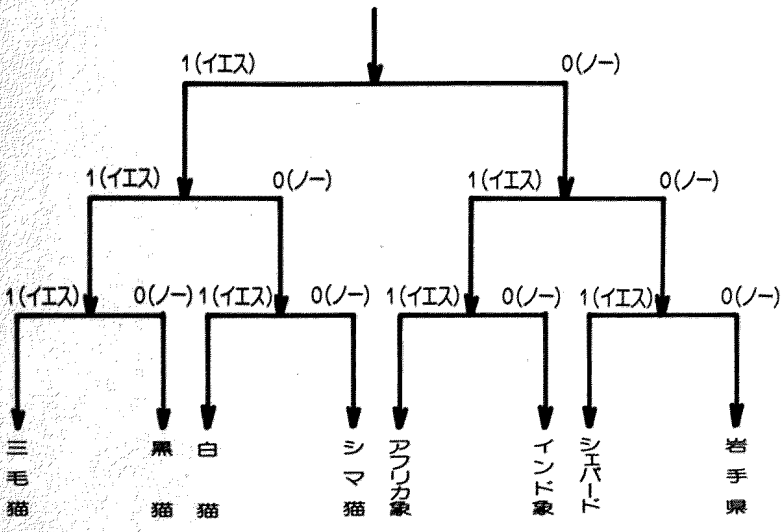


図2－5　情報をイエスとノーの組合わせだけで表わす

変えることによって、いろいろなことがらを表わすことができます。例えば図2―4の6番と7番の質問を「大きいか？」「黒か？」に変えて、右側の太線は変えないとすると、この太線の最後の点、つまりイエス・イエス・イエス・ノー・イエス・イエス・イエスという組合わせは、大きな黒猫を表わすことになるわけです。質問が変わるたびに表わすものが変わるのでは困るので、ちょっと工夫しなければなりません。

要するに、区別することができればいいわけですから、図2―5のように、最後に到達した点一つ一つに、区別したいものごとを対応させておけば、質問はいらなくなります。

図2―5では、イエス・イエス・ノーの組合わせ、すなわち1・1・0の組合わせは黒猫を表わし、0・1・0はインド象を表わす

ことになります。このトーナメント図のようなものの段数を増やせば、より多くのものを1と0の組合わせで表わすことができます。最後の到達点それぞれに、必要な事柄をあらかじめ対応させておけば、1と0だけでできている数字の組合わせを人間の言葉のように使うことができます。

コンピュータはこの1と0の組合わせで情報を取り扱っています。もちろん、その組合わせと事柄との対応は、人間があらかじめ決めておきます。

1と0の組合わせといえば、2進法の数字とよく似ています。図2—5の、1・1・0とか、0・1・0とかの組合わせを、110、010、というふうに書けば、2進数とまったく同じです。ただ、2進数で、10と書けばいいところを、わざわざ010と書くところが違うだけです。情報としての2進数は、いわば番号だからです。切符の通し番号は10進数ですが、0082とか0130のような番号の書き方をします。それと同じことです。

コンピュータでは、情報を2進数に置きかえて扱っているのです。その2進数の一桁には、0か1か、二通りの数字を置くことができるので、2進数の一桁分で、二つの事柄を伝えることができます。二つある事柄のうち、どちらか一方であることを伝える情報を、1ビットの情報といいます。これがビットの意味です。いいかえれば、2進数一桁分で表わせる情報量を1ビットというのです。

ある一つの質問にイエスかノーで答えると、その答えは1ビットの情報だといえます。

2ビット、3ビットとは、それぞれ、2進数の二桁分、三桁分で表わせる情報量のこと

ですが、2進数二桁分ではどういうことが表わせるかというと、00，01，10，11，の計四つの事柄のうち一つを伝える情報が表わせるのです。三桁分では、000，001，010，011，100，101，110，111の計八つの事柄のうちのどれか一つを伝える情報が表わせることになります。（図2—5は3ビットの情報の仕組みを表わしています。）

このように、ビット数が一つ増すごとに、その情報の伝えうる事柄の数は倍々になって行きます。4ビットでは、一六の事柄を伝えることができますし、8ビットでは、$2^8=二五六$の事柄を伝えることができます。

4ビットのことを1ニブル(nibble)、8ビットのことを1バイト(byte)ともいいます。12個を1ダース、12ダースを1グロスといったりするのに似ています。

コンピュータが、入ってくる情報を何ビットの情報として処理するかで、コンピュータの性能に違いがでてきます。普通、マイコンでは、8ビット型が多いようです。ほかに、10進法の計算に適した4ビット型、ミニコン（マイコンより少し大きなコンピュータ）に近い、12ビット型、16ビット型があります。

4ビット型では4ビット＝1ニブルを、8ビット型では8ビット＝1バイトを、12ビット型、16ビット型ではそれぞれ12ビット、16ビットを、そのマイコンの語長といいます。コンピュータが、一度に扱える情報量（何ビットの情報かということ）を、人間の言葉のように、語と呼びます。

4ビット型では、0000～1111の2進数16個を使って、情報を表わし、8ビット型では、00000000～11111111の2進数二五六個を使っ

4ビット型マイコンの使える語（1語4ビット）

16個

0000, 0001, 0010, ……, 1110, 1111

語長

8ビット型マイコンの使える語（1語8ビット）

256個

00000000, 00000001, ……, 11111111

語長

図2－6　マイコンの使える語

て、情報を表わしています。だから、語長が長いほど、複雑な仕事をするのに向いているといえます。（図2－6）

マイコン内部では、2進数表現の情報を並列に処理しています。

並列処理とは、8ビット表現の情報なら、8桁分の数を一度にやりとりすることです。これに対して、直列処理とは、一桁、一桁を

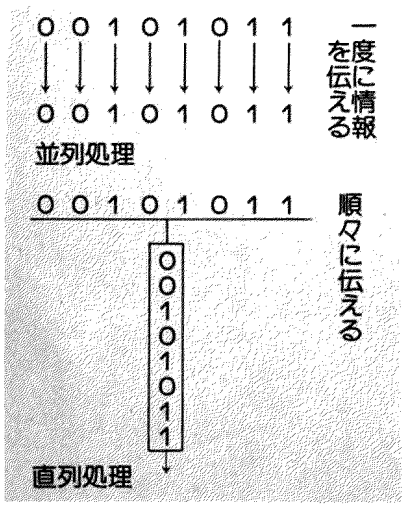


図2－7　並列処理と直列処理

順々に伝えて行くやり方です。（図2―7）

マイコンの基本的機能

マイコンは、大きくわけて三つの部分に分かれているとまえに書きましたが、その三つの部分を、さらに五つの基本的な働きに分けることができます。

マイコンには五つの働きがある

五つの基本的な働きとは、マイコン全体を監督する「**制御**」、計算や判断をする「**演算**」、いろいろなことを記録しておく「**記憶**」、マイコンの外から情報を取り込む「**入力**」、マイコンの外へ情報を伝える「**出力**」の五つです。

これは、マイコンに限らず、どんなコンピュータにもいえることです。

これらの五つの働きの間を、2進法で表わされた情報が走りまわって、マイコンがいろいろな仕事を実際に行うのです。

実は、このマイコンの仕組みは、私達がものを考える時の仕組みによく似ています。マイコンの働いている場面を見ると、まるでマイコンが「ものを考えている」ように見えます。確かに、マイコンは人間と同じ仕組みで「考える」のです。

それで、マイコンをも含めて、コンピュータのことを、「人工知能」と呼ぶわけです。いわば、コンピュータは、人間に作られた考える機械なのです。

ですから、五つの基本的な働きを、人間にあてはめてみることができます。

例えば、あなたが誰かに、

「3と9を足すといくつでしょう」

という問題を出されたとします。たちどころに、12だとわかり、

「12です」

と答えるでしょうが、実は、その一瞬の間に、あなたはこの五つの働きをしているのです。

今のことを、もう少し細かく分けてみましょう。

(1) 「3足す9は?」という問題を耳でき く。

(2) 「3足す9は?」という問題が出されたことを覚えておく。

(3) 3に9を足して、12という答えが出る。

(4) 「12です」と答える。

この(1)にあたる働きが入力です。「3足す9は?」という質問は、いろいろな情報を与えてくれます。まず、二つの数を足すのだということ、次に、その二つの数とは3と9であること、そして、こうきかれた以上、その答えをいわなければならないこと。

このように、行わなければならないこと(この場合、「足す」ことと「答える」こと)や、それを行うために必要な数字(この場合は3と9)などの情報を受け取って頭に送ることを入力というのです。

(2)にあたる働きは、記憶です。「3足す9は?」という情報は、音として耳から入るのですが、音というものは、一度ものごとを伝えたら、それっきりどこかに行ってしまいます。ですから、そういう質問がされたのだということを覚えておかなければなりません。「足す」、「答える」、「3と9」というような情報を覚えておくこと、それが記憶です。

(3)にあたる働きは、演算です。普通は、こ

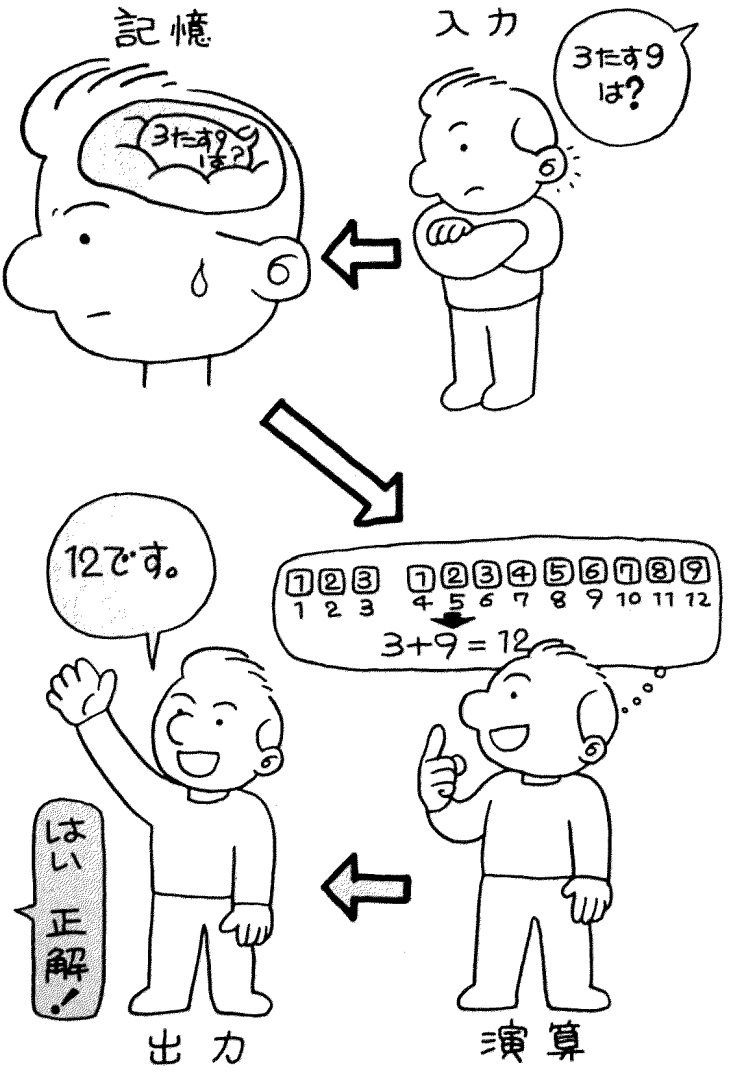


図2－8　入力、記憶、演算、出力とは

のくらい簡単な足し算は、その答えも覚えているものですが、小さい頃は、そういうわけにもいかなかったでしょう。

まず、三つ数えて、さらに九つ数えて、指を折りながら考えたはずです。

あるいは、3から1借りてきて、その1を9に足して10、3から1借りたので残りは2、10と2で12、というように考えたりもしたでしょう。

このような、問題の計算などを演算といいます。もし、質問が「3と9とどちらが大きいか」だったら、その3と9を比べて、9の方が大きいと判断することも、この演算という働きにあたります。

(4)にあたる働きが出力です。計算して出てきた答えを、頭の中で思っていただけでは答えたことになりません。答えを口でいったり、紙に書いたり、手で示したり、どういうふうに伝えるにせよ、答えの情報を外に出さなければなりません。この答を出すことを出力というのです。

あと一つ働きが残っています。

(5)制御です。

制御という言葉は、英語の方がわかりやすいかも知れません。制御とは、コントロールのことです。ここでは、ほかの働きがうまく行えるように調整したり、命令したりすることです。

質問されても、それを聞こうとしなかったり、つい聞きそびれたりすれば、その質問に答えることはできません。問題を考えまい、考えまいと頑張っていれば答えは出てきません。もし答えが出ても、絶対に口に出さないぞと頑張っていれば、答えは人に伝わりませ

ん。

制御とは、それぞれの働きが、そのように勝手な行動をしないためにコントロールするものです。

また、演算よりも先に出力したり、入力より先に記憶したりしては、おはなしにもなりません。制御は、まず入力、次に記憶、そして演算、それから出力というように順序正しく、それぞれの働きが行われるように、コントロールすることでもあります。

これら五つの働きが、マイコンに限らず、どのコンピュータにも共通した基本的な働きです。

もう一度、もっと大がかりな例で、同じことを表わしてみましょう。

ある探偵会社があるとします（普通は、探偵会社のことを興信所といいます）。現実の探偵は、シャーロック・ホームズや多羅尾伴内のように、難事件の解決を一人で行ったりするわけではなく、たいていは、誰かの身辺調査などをしています。会社の形になっていれば、社長がいて、その下に平社員がいるわけです。

探偵会社に、A氏が、B氏の身辺調査を依頼したとします。

まず、A氏は身辺調査をしてほしい相手がB氏であること、B氏とは、どこに住んで、何を職業にして、どんな顔をしているか、などA氏がB氏について知っていることを、その会社に伝えます。これが入力です。

受け付けた係の人は、A氏から聞いたことを書類にします。これが記憶です。

その書類を、社長に持って行くと、社長はその調査を社員の探偵に命令します。これが

制御です。また、社長は、受け付け係の人がさぼっていないかとか、ほかの社員がちゃんと仕事をしているかとか、監督しているわけで、これも制御の働きの一つです。

そうして、社員の探偵は、B氏の調査を実行します。

B氏の勤め先の人に話を聞いたり、B氏の近所の人に聞いたり、B氏を尾行して証拠写真を撮ったり、資料の収集のために動きまわります。これらの資料も、次々に書類の形で「記憶」されて行きます。

この資料集めも、入力ということになりますが、コンピュータでは計算問題などの時、入力はたいてい一度しか行いません。その一度の入力の時に、必要な情報はすべて入れてしまうのが普通です。しかし、ゲームなどをコンピュータで行う時は、ハンドルやスイッチを、その時、その時に動かして遊びます。ハンドルやスイッチを動かすことも、入力の一つですから、その場合は、何度も入力が行われることになります。入力が行われれば、必ず、まず記憶が行われます。

さて、探偵の話にもどります。探偵は、あちこちから集めたB氏の資料を会社に持ち帰って、たくさんのばらばらの資料から、B氏がどういう人であるか、判断します。これが演算です。

演算という言葉は、単に数字の計算だけを意味するのではなく、このように判断することも含まれています。大小の比較なども、演算です。

コンピュータの中では、あらゆる情報がすべて2進数で表わされているので、判断するという働きも、数字の操作になっています。

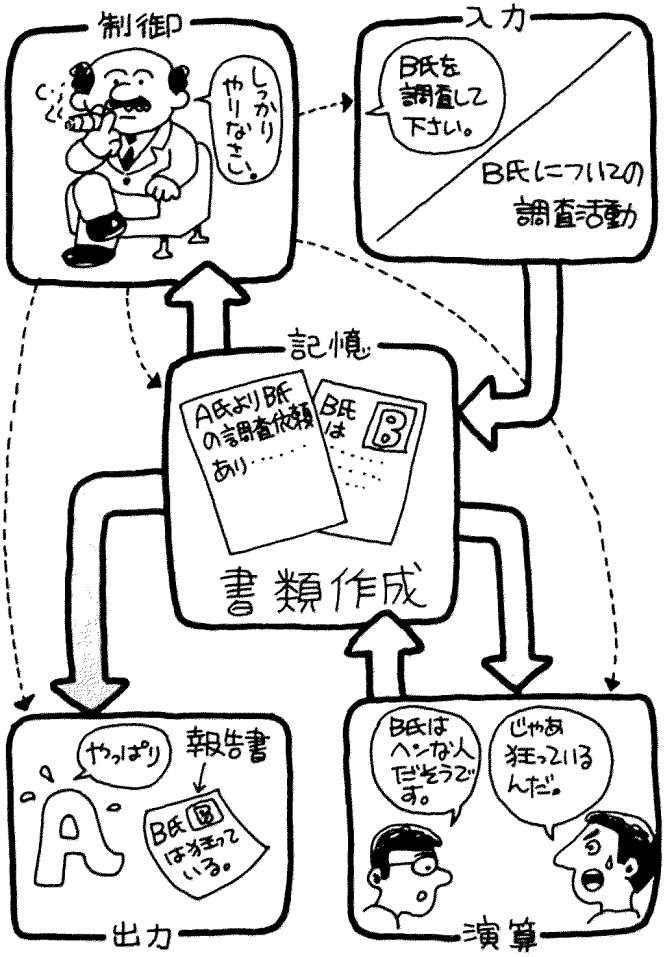

制御
しっかりやりなさい。
入力
B氏を調査して下さい。
B氏についての調査活動
記憶
A氏よりB氏の調査依頼あり……
B氏は……
書類作成
出力
やっぱり
報告書
B氏はヘンな人だそうです。
演算
B氏はヘンな人だそうです。
じゃあ狂っているんだ。

図2－9　マイコンの働き

ですから、見かけは、すべての仕事が計算のように行われます。

B氏がどういう人であるか結論が出たら、その結果を報告書にまとめます。これも記憶です。コンピュータでは入力のすぐあとに必ず記憶の働きを行い、出力のすぐまえにもやはり記憶の働きを行います。演算の前後、制御のまえも同じです。

さて、B氏の身辺調査報告書ができあがれば、A氏がやってきて、それを受け取ります。これが出力です。

以上の事柄を、今度はコンピュータで説明しましょう。

マイコンの構造

入力される情報は、コンピュータでは、プログラムとか、データとかいう名前で呼ばれます。

プログラムとは、行わなければならないことの情報です。今までの例では、「足すこと」、「答えること」、「身辺調査をすること」などの情報です。

図2—9では、「身辺調査の依頼があった」という情報が社長に行って、社長はその情報に基づいて、制御を行うわけですが、コンピュータでも、命令の集まりであるプログラムの情報は、制御の働きを行う部分に行って、制御するところは、その情報に基づいて各部分に命令を発するのです。

この命令を、制御信号といいます。図2—9では、社長からでている点線がそれにあたります。

データとは、今までの例では、「3と9」と

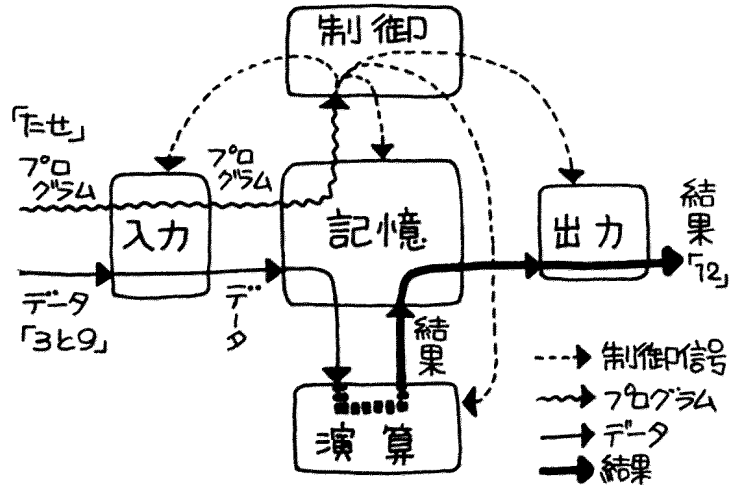


図2－10　コンピュータの基本的機能

か、「B氏はへんな人だ」などの、計算を行う具体的な数値や、判断の材料、資料などのことです。データは、日本語に訳す時は、たいてい「資料」と訳されます。コンピュータ用語としては、データという英語しか使いません。

　このプログラムとデータが入力され、プログラムの情報は制御の部分へ、データの情報は演算の部分へ行き、決められた仕事を行ったのち、出てきた結果を出力するのです。

　この五つの働きを、マイコンでは三つの部分で分担しています。

　CPUは、制御と演算、それから少し記憶の働きも受け持っています。CPUが、マイコンの心臓部といわれるのは、制御と演算という、もっとも大切な役割を受け持っているからです。マイコンでは、CPUはたった一

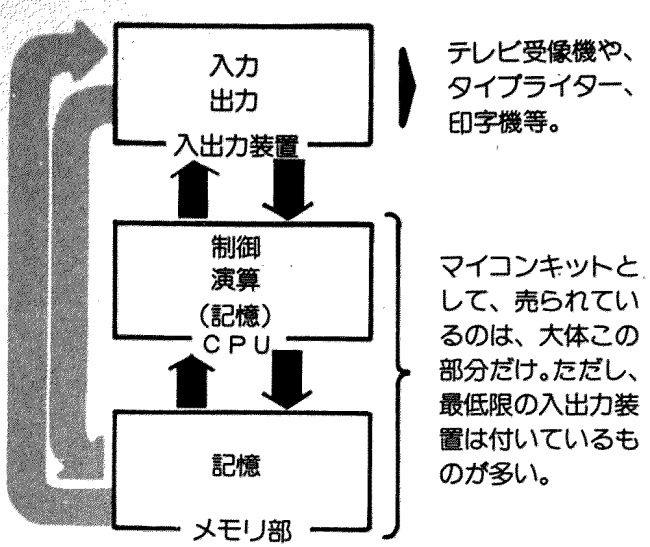


図2－11　マイコンの基本的機能

個のＬＳＩでできています。ＣＰＵになる特別なＬＳＩのことを、マイクロ・プロセッサとか、ＭＰＵ（マイクロ・プロセシング・ユニット）などと呼ぶことがあります。

メモリ部は、名前のメモリ（記憶という意味の英語）で表わされているように、記憶の働きをするところです。マイコンでは、メモリ部は普通何個かのＩＣでできています。

マイコン・キットとして売られているものは、通常、このＣＰＵとメモリだけです。最低限の入力と出力を行う装置も付いているのが普通ですが、十分な働きをさせるためには、いろいろな入出力装置を取り付けなければなりません。

入出力装置は、文字通り、入力と出力を行う装置です。入力装置と出力装置が別々になっていることも、一つの機械に入っているこ

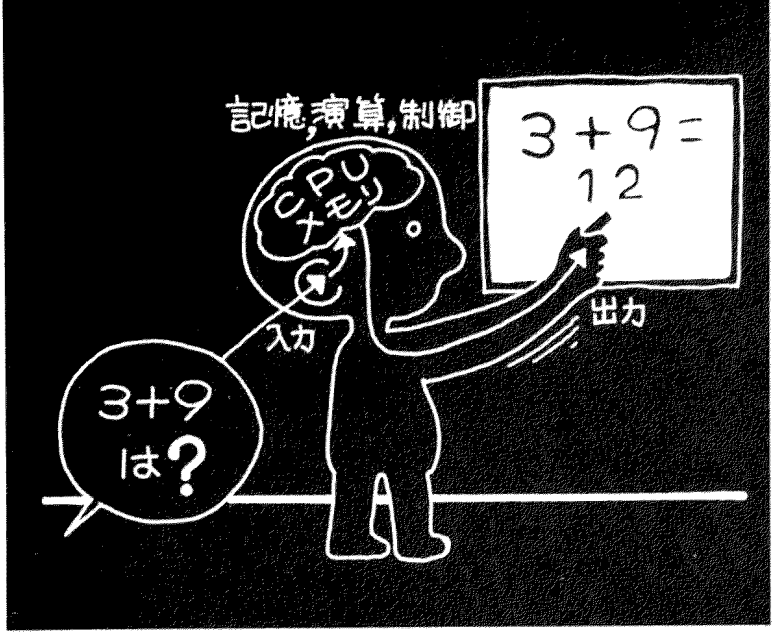


図2－12　マイコンを人間にたとえれば

ともありますが、これらをまとめて、入出力装置と呼んでいます。また、入力という意味のインプット、出力という意味のアウトプットの頭文字をとって、**I／O装置**（アイ・オー装置）と呼ぶこともあります。I／Oと書けば、入力／出力の意味で、よく使うので覚えておいた方がよいでしょう。

これら三つの部分、CPU、メモリ部、入出力装置が互いに働きあって、いろいろな働きをしてくれるわけです。

この三つの要素の関係は、図2—11のようになっています。図2—10と見比べてみればよくわかるでしょう。

また人間にたとえれば、CPUとメモリはちょうど頭にあたります。入出力装置は、目、耳、口、手、足、などにあたります。

CPU、メモリ、入出力装置について、そ

れぞれ、もう少し詳しく説明しましょう。

CPUとは

Central Processing Unit

中央処理装置

CPUは、普通、コンピュータの心臓部といわれています。ここで、コンピュータ全体のコントロール、データの計算や判断（大小の比較等）などをつかさどっています。

マイコンでは、大体、一個のLSIがCPUを形成しています。このLSIを、マイクロ・プロセッサともいうことは、まえに書きました。

CPUの中は、大きく分けて演算論理回路（ALU）、制御回路、レジスタ部の三つで構成されています。

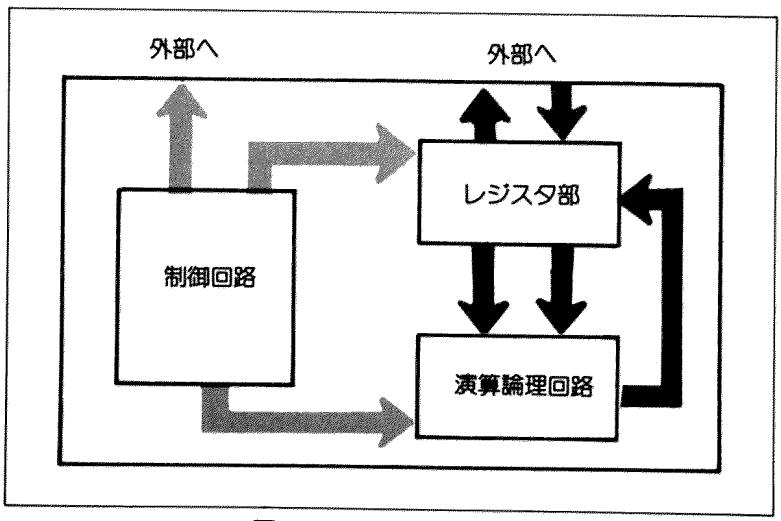


図2－13　CPUの内部

演算論理回路

ここは、データの計算、判断などをする部分です。

コンピュータは、基本的には足し算しかできないとまえに書きました。掛け算は、足し算の繰返しで計算できますが、引き算は補数というものを使って、足し算のように計算します。割算は引き算の繰返しです。

4ビット処理の場合を例にとって、補数による引き算の仕組みを説明しましょう。

(1) 補数

Aという数からBという数を引く場合、Bを補数に変換してAに足すと、A引くBの答えが得られる仕組みになっています。

例えばBが4ビット表現の情報だとして、

B＝0010

なら、Bの補数Cは、

C＝1110

になります。

A＝1011として、A＋Cを計算してみると

```
  1011…(A)
+)1110…(C)
 11001
```

A＋C＝11001→1001

となりますが、4ビット処理ですから、5桁目の1は無視されて、A＋C＝1001という答えが出てきます。ところで、A－Bは

```
  1011…(A)
-)0010…(B)
 1001
```

A－B＝1001

となり、結果的にA－B＝A＋Cということになるのです。

補数という名前は、CがBを補って0になるというところからきています。

B＋Cを計算すると、

0010……（B）
＋）1110……（C）
10000

B＋C＝10000→0000
（4ビットなので五桁目は切り捨てる）

となります。逆にいえば、補数はこのような性質を持つように決められるということです。しかし、もちろん人間が命令を与える時は、A－Bを計算せよというふうに命令します。それをコンピュータが－Bの部分を、＋Cに変換して計算するのです。

(2) 論理回路

計算などの働きをする回路を論理回路といいます。演算論理回路は、この論理回路をたくさん組合わせてできています。

論理回路には、基本的なものが三種類あります。論理回路の論理は、論理学の論理です。論理学とは、真（真実、真理の真）と偽（偽物の偽）とを取り扱う学問です。この真に1、偽に0を対応させて、論理学が真偽を取り扱うのと同じように、論理回路が1、0を取り扱うのです。

① アンド回路（論理積の回路）

アンド（AND）は、日本語の「かつ」の意味です。AアンドBは、AかつBのことで、Aであり、そのうえBでもあるという意味になります。アンド回路は、論理積回路ともいい、数学の集合で使う積と同じ意味です。

論理学では真理表というものを用います。真理表とは、A、Bの真理値（真か偽かの値。コンピュータでは、1か0かの値）によって、AアンドB（論理代数学の記号では、A・Bと書きます。・は、数学で積の記号に用いることがあります）の真理値がどのようになっているかを表にしたものです。

表2—3に示すように、論理積では、どちらとも真（1）でなければA・Bは真（1）になりません。

図2—14にアンド回路の簡単なものを示しました。A、Bのスイッチが同時にオンにならなければ電球はオンになりません。コンピュータでは、オンを1、オフを0に対応させますから、いいかえるとA、Bが1でなければ電球を流れる電流も1にならないということになります。電球がつくかわりに、オンの

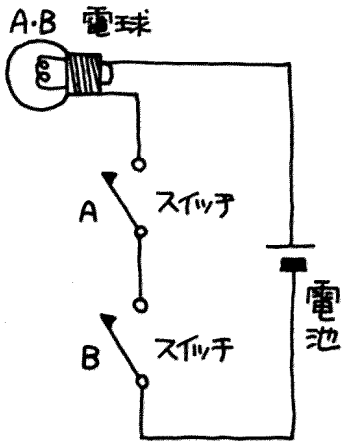


図2—14　簡単なアンド回路

A	B	AアンドB（A・B）
偽（0）	偽（0）	偽　（0）
偽（0）	真（1）	偽　（0）
真（1）	偽（0）	偽　（0）
真（1）	真（1）	真　（1）

表2—3　A・Bの真理表

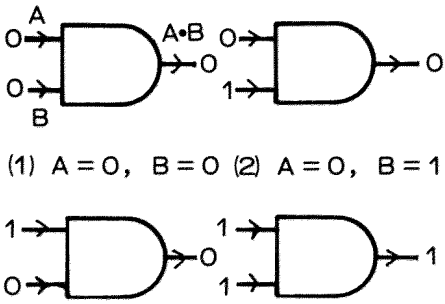


図2－16　アンド回路の働き

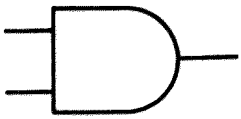


図2－15
アンド回路のシンボル
（MIL規格）

信号（1の信号）が出てくることにすれば、コンピュータのアンド回路と原理的には同じです。

アンド回路は図2―15のようなシンボルで普通表わされます。このシンボルを使って表2―3を表わしたのが図2―16です。

図2―16で、0というのは電気信号がないこと（オフ）を示し、1は電気信号がある（オン）ことを示しています。A、Bともに電気信号のある（オン）時だけ、右側に電気信号が伝えられます。つまりオンになります。

② オア回路（論理和の回路）

オア（OR）とは、「または」、「あるいは」という意味です。オア回路を論理和回路ともいいます。

論理和は、AかBか、少なくともどちらか一方が真（1）であれば、AオアB（論理代

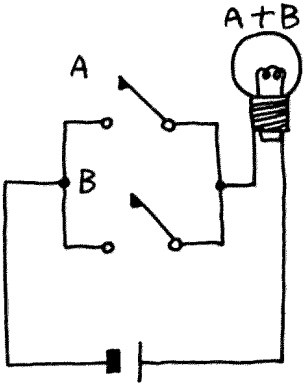


図２－17　簡単なオア回路

A	B	AオアB (A＋B)
偽 (0)	偽 (0)	偽 (0)
偽 (0)	真 (1)	真 (1)
真 (1)	偽 (0)	真 (1)
真 (1)	真 (1)	真 (1)

表２－４　Ａ＋Ｂの真理表

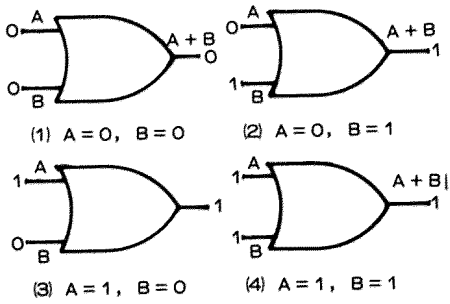


図２－19　オア回路の働き

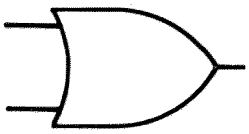


図２－18
オア回路のシンボル
（ＭＩＬ規格）

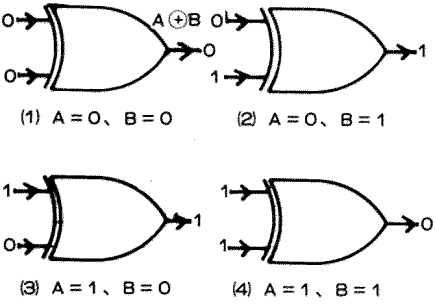


図２－21　エクスクルシブ・オア回路の働き

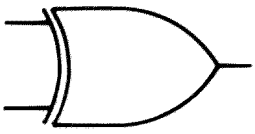


図２－20
エクスクルシブ・オア回路のシンボル
（ＭＩＬ規格）

数学の記号で、A＋B。数学では和の記号に用いています）も真（1）になることを意味しています。

表２―４、図２―17～19は、アンド回路の時と同じように見ます。

オア回路には、もう一つ、エクスクルシブ・オア（排他的論理和）回路というものがあります。エクスクルシブ（exclusive）とは、他を排すること、つまり他との共存を許さないという意味です。人間の関係でいえば、仲

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	⓪

ここが、ただのオアとちがう。

表２－５　A⊕Bの真理表

の悪いA君とB君は、同じ部屋に一緒にいることがないという状況にたとえられます。

エクスクルシブ・オアは、記号で⊕と書きます。

③ ノット回路（否定回路）

ノット（NOT）とは、日本語でいえば「～でない」という意味です。

否定を表わす論理記号は$\bar{A}$（ノットエイと読む）を使います。真の否定は偽、偽の否定は真です。1、0でいえば、1の否定は0、

A	ノットA（$\bar{A}$）
0	1
1	0

表2－6　$\bar{A}$の真理表

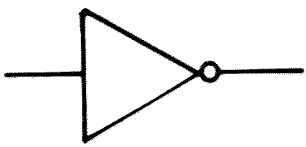
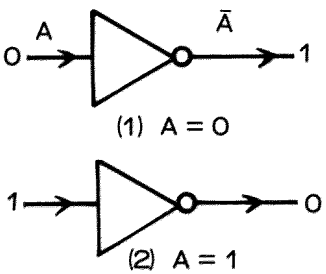
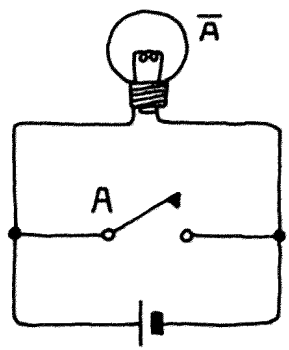


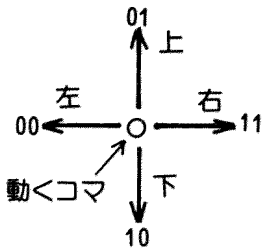
図2－22①　ノット回路のシンボル（MIL規格）





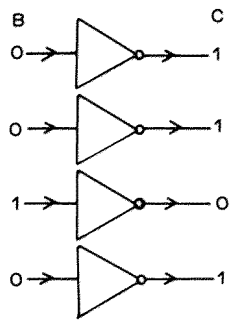
Aのスイッチがオン(1)になると，Aの方に全て電流が流れる(ショートする)ので電球がつかない。(オフになる)

図2－22③　ノット回路の働き　図2－22②　簡単なノット回路



今、上下左右どちらの方向に動いていても、逆方向に動かすためには、今動いている方向を示す2進数の補数の方向へ動けと命令するだけでよい。
00(左)の補数＝11(右)
10(下)の補数＝01(上)

図2－24



0010……(B)
＋) 1101……(C)
1111

図2－23

0の否定は1です。

ところで、前に述べた補数には、もう一種類あって、B＋C＝1111となるようなCを、Bの補数ということがあります。このような補数は、このノット回路によって簡単に得られます。

B＝0010

として、Bの補数Cの出てくる仕組みを図2―23に示します。

数計算をする時は、前に述べた補数（これを2の補数といいます）を用います。ここで述べた補数（1の補数といいます）は別の使い方をします。

例えば、マイコンでテレビゲームをする場合、コマの進む方向を2進数で表わしておいて、動きを命令するのですが、図2―24のような場合、逆方向に進めと命令するために、

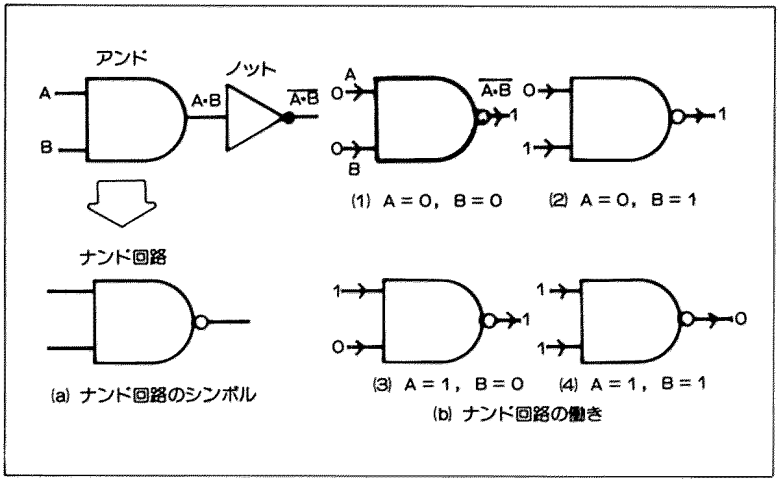


図2－25　ナンド回路

1の補数をとらせると便利なのです。

④　①②③の複合

アンド、オア、ノット回路を組合わせてナンド回路(NAND)、ノア回路(NOR)ができます。

ナンド回路は、ノット回路とアンド回路の組合わせ、ノア回路はノット回路とオア回路の組合わせです。それぞれ、図2―25、図2―26のような働きをします。

コンピュータに使うICでは、ナンド回路で構成される部分が多いといわれています。

CPUとなるマイクロ・プロセッサでは、トランジスタによる論理回路が数百から数千個組合わされています。この論理回路によって、情報が伝えられ、受け取られ、変換されて、仕事を行っているのです。計算機能だけでなく、ほかにも1、0の情報の伝達、操作

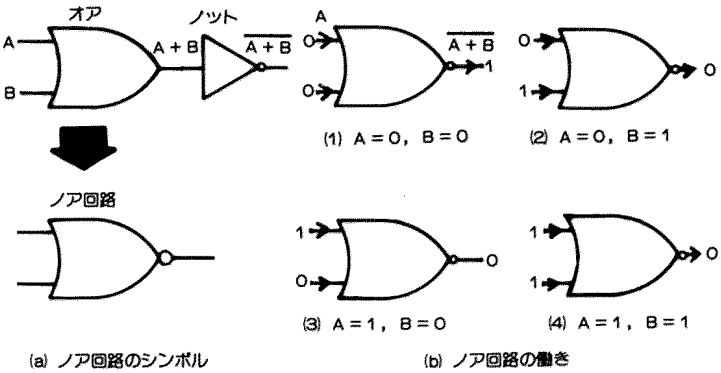


図2－26　ノア回路

に関するいろいろな働きをしています。例えば記憶する部分なども、この論理回路の組合わせです。これについては、あとで述べます。

⑤ 加算回路

CPUのALU（演算論理回路）は、ほぼ加算回路（足し算をする回路）を中心に構成されています。加算回路も、論理回路の組合わせです。

A＝0010，B＝0110，として、図2—27に加算回路の基本的な仕組みを説明しています。

Aの一桁目、二桁目、三桁目、四桁目の数字と、Bの一桁目、二桁目、三桁目、四桁目の数字が、それぞれ桁ごとに足し合わされ、右の方に答えが出てきます。上の回路に接続された線は、繰り上げを表わしています。

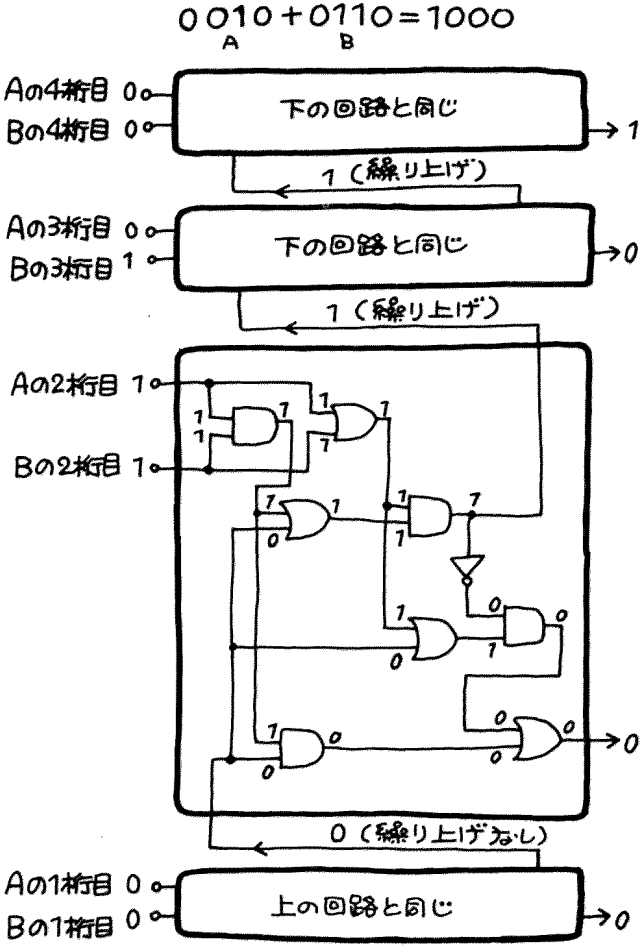


図2－27　加算回路の基本的な仕組み

制御回路

マイコンが命令されたことを忠実に行うには、その命令を受け取って、マイコン各部に指令を発する装置が必要です。いわば、マイコン各部の社長のようなものです。

制御回路は、命令レジスタ、命令デコーダ、制御信号発生回路の三つの部分でできています。入ってきた命令を一時蓄えておく部分が命令レジスタです。レジスタ(register)とは一般に、情報を一時蓄えておく部分の名前です。しかし、一語から数語の情報しか記憶しないので後で述べるメモリとは区別されます。

命令レジスタも、一つの命令しか記憶しません。命令レジスタは入ってきた命令情報を命令デコーダに渡して、次の命令をまた一つ記憶しておきます。

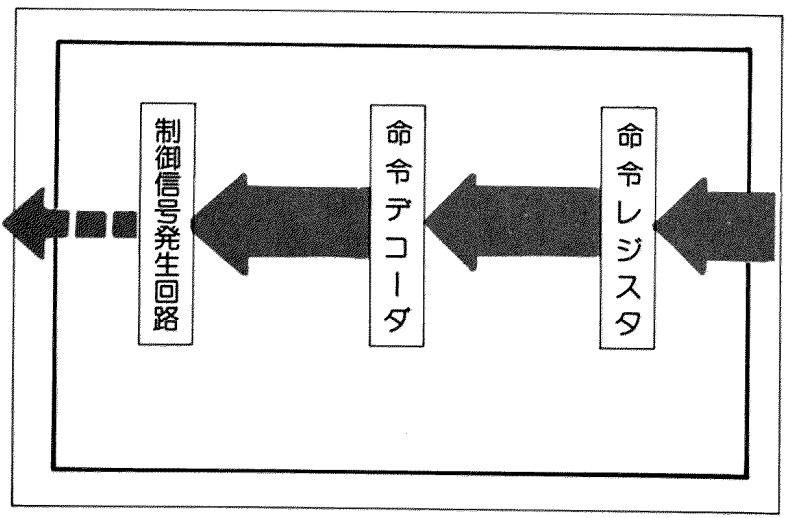


図2－28　制御回路

命令デコーダは、その命令情報の意味を解読して、制御信号発生回路に送ります。

制御信号発生回路は、それに従って制御信号を、マイコンの各部に送ります。

マイコンの各部は、この制御信号に従って、記憶したり、情報を送り出したり、計算したりします。

制御信号には、クロック・パルスというものが加わって、マイコンの時計としての役割を果たします。このクロック・パルスに従って、マイコン各部はタイミングよく作動するのです。大体、クリスタル（水晶）を使って正確なクロック・パルスを作っています。

レジスタ部

レジスタとは、命令レジスタのところで書いたように、少量の記憶を一時的にするものです。

たくさん本を持っている人が、何かについて執筆する時、何冊か必要な本を本棚から取り出して、参照しながら書き進めて行くのが普通です。

それと同じで、レジスタは、メモリ部から引き出した情報を、演算論理回路や制御回路のために近くに置いておく働きをします。

レジスタには、いろいろな目的に使える汎（はん）用レジスタのほかに、特定の働きをするものがあります。

ＡＬＵ（演算論理回路）にくっついて、演算を行うデータを一時記憶したり、その結果を一時記憶しておくアキュムレータ。ＣＰＵが次に実行する命令の記憶されている場所を常に示しているプログラム・カウンタ。データの記憶されている場所を示しているデータ

・カウンタなどです。

メモリ部

CPUが仕事をするためには、その仕事の情報が必要です。仕事の手順（プログラム）やデータなど、人間の側から見れば単純そうな仕事でも、コンピュータにしてみれば多くの情報が必要です。その情報を記憶する回路をCPUに組込むのは少し無理な話です。もっとも、第一章にあげた、ワンチップ・コンピュータでは、このメモリも、後で述べるインタフェースも組込んでいるのですが、それでは汎用性（いろいろに使えること）に欠けます。それで多くのマイコンでは、記憶するためのICを幾つか、CPUのまわりに取り付けてメモリ部としています。

メモリとは、記憶用ICの総称です。英語でメモリ（memory）といえば「記憶」のことです。

メモリの中は、数千個のメモリ・セルでできています。メモリ・セルとは、1ビットの情報を記憶する回路で、このメモリ・セルを何個か並べて、一語（8ビット型では一語が8ビット＝1バイトであることはまえに述べました）を記憶しているのです。

メモリには、大別して2種類あります。読み出すだけで、記憶内容を変更できないリード・オンリ・メモリ（Read Only Memory 略してROM）と、プログラムによって記憶内容を自由に変更できるリード・ライト・メモリ（Read Write Memory 略してRWM）の二つです。RWMは、ランダム・アクセスメモリ（Random Access Memory，略して、R

AM）ともいい、普通、RAMという呼称を用いることが多いようです。

アドレス線とデータ線

メモリに書き込まれている情報を必要に応じて呼び出す役割を果たすものに、アドレス線とデータ線があります。

アドレス（address）とは、住所という意味の言葉で、コンピュータでは、コンピュータ各部の場所を番号付けした番地のことです。アドレス信号は、8ビット型のマイコンでは、普通16ビットの情報です。これは、アドレスが最終的にはメモリ内のメモリ・セルまでを指定するために、数千以上の番地を表わす必要があるからです。16ビットだと、

$2^{16}=65536$

ですから、単純には六万個以上の番地を表現できますが、実際には表現上の制約で、それほどのことはできません。

アドレス情報は、16桁の2進数でできているわけですが、その上の方の桁で、まずどのLSIかを指定し、下の桁で、そのLSIの内部のどの部分かを指定します。ここでいうLSIは、メモリ以外のLSIも含みます。このアドレス情報によって、今出ている制御信号が一体どこに向けて発せられたものかを各LSIは知るわけです。

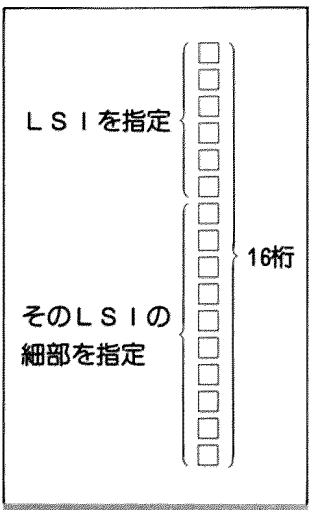


図2－29　アドレス情報

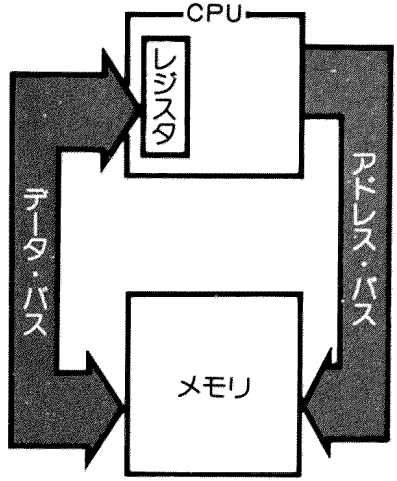


図2－30

このアドレス情報は、メモリ部においてもっとも複雑な情報を伝えます。

CPUから、アドレス情報が伝わって、メモリ内の特定のメモリ・セルの列を指定すると、そのメモリ・セルの列に蓄えられていた情報が、データ線を通ってCPUに取り込まれるのです。メモリ・セルとは、1ビットの情報を記憶する記憶細胞のことです。

また、逆にアドレスによって指定されたメモリ・セルの列が、CPUから送られてくる情報を記憶することもあります。

8ビット並列処理のマイコンの場合、データ情報は8ビットですから、これを並列に処理するためには8本のデータ線が必要です。アドレス線も同じ理由で16本必要です。マイコンでは、これをまとめて、それぞれデータ・バス、アドレス・バスとしています。バスとは、町中を走るバスと同じ言葉です。

ここでいうデータは、メモリに記憶されている内容のことで、プログラム情報も含みます。

ROM

ROMには、大別して二種類あります。

製造時に記憶内容が決定されていて変更の

まったくできないものをマスクROM（MROMと略記する）といい、製造時は、全く記憶内容を持っていなくて、あとでそれを使用する人が便利なように、一つ一つ内容を書き込むものをプログラマブルROM（プログラム可能なROM。PROMと略記）といいます。

大抵のマイコン・キットにはMROMが実装されていて、これにコンピュータとして作動するために必要な基本的プログラム（モニタ・プログラム）や、アセンブリ言語を機械語に翻訳するためのプログラムなどが書き込まれています。

PROMは、マイコン・キットにはあまり用いませんが、機械組込み用のマイコンなどではよく使います。

PROMには、幾種類かあって、一度書き込んだら二度と消せないものと、再び消して書き込めるものとがあり、消す方式も、紫外線照射によるものや、電気的処理によるものがあります。

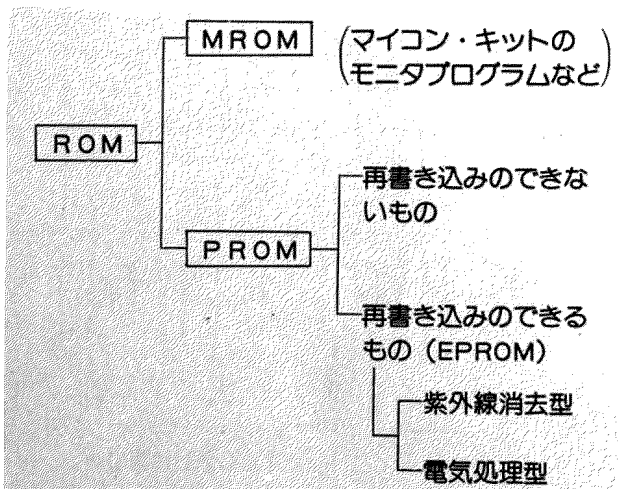


図2－31　ROMの種類

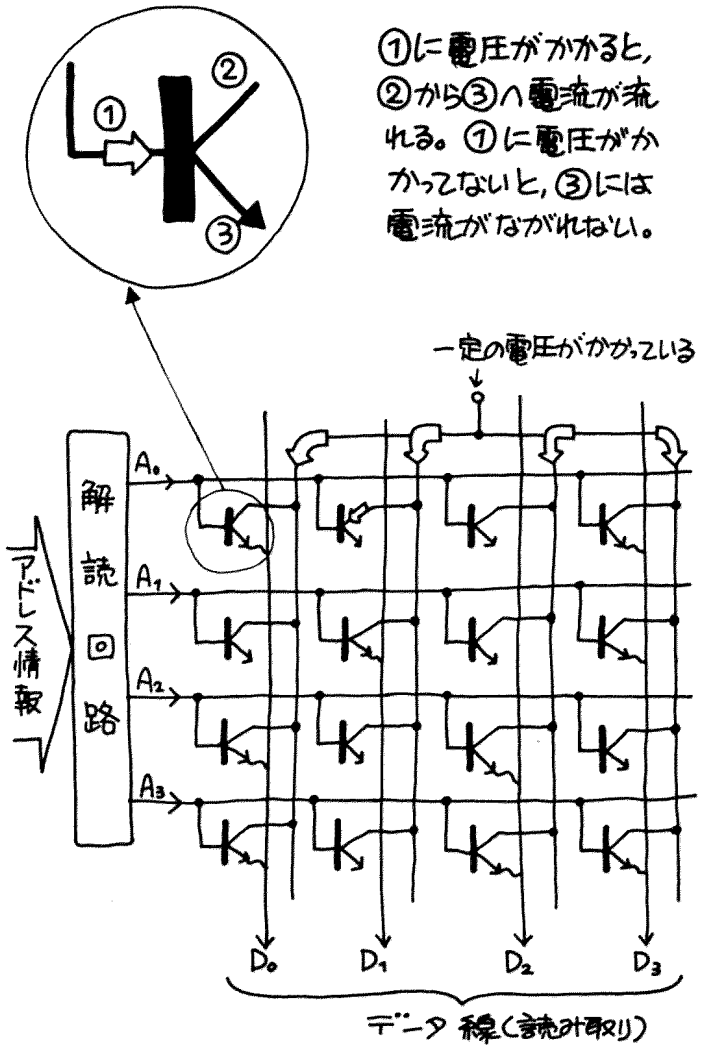


図2－32　ROMの簡単な構成

ROMのメモリ・セルは、読み出すだけでいいので、RAMに比べれば少し簡単な構造を持っています。4ビットの情報を四語記憶しているROMを例にして、その原理を簡単に説明します。

図2―32中の、丸囲みの部分は、トランジスタの記号です。①に一定レベル以上の電圧がかかると、②から③へ電流が流れる仕組みになっています。①に電圧がかかっていないと②から③へは電流が流れません。

図の左から、アドレス情報が入って、解読回路でその情報が、A_0～A_3の、どの番地を指定するかを解読して、それが例えばA_2ならA_2に電圧をかけます。

すると、A_2上に接続しているメモリ・セル四個の、それぞれのトランジスタに電圧がかかります。図2―32の①の部分にかかるわけ

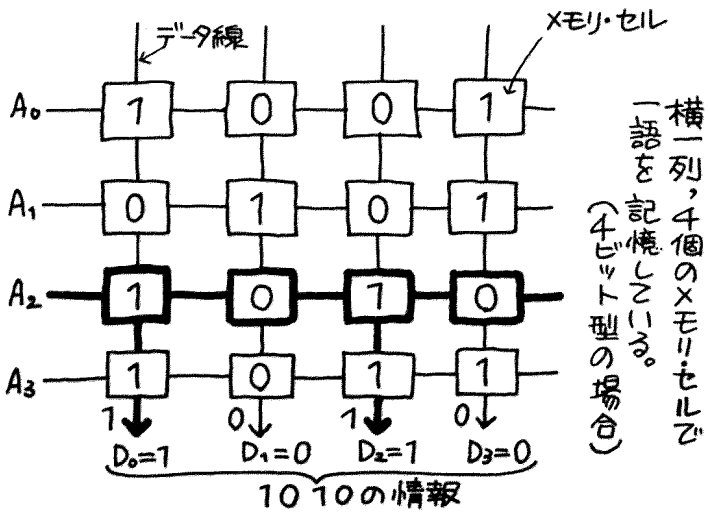


図2－33　図2－32でA_2が選ばれた場合

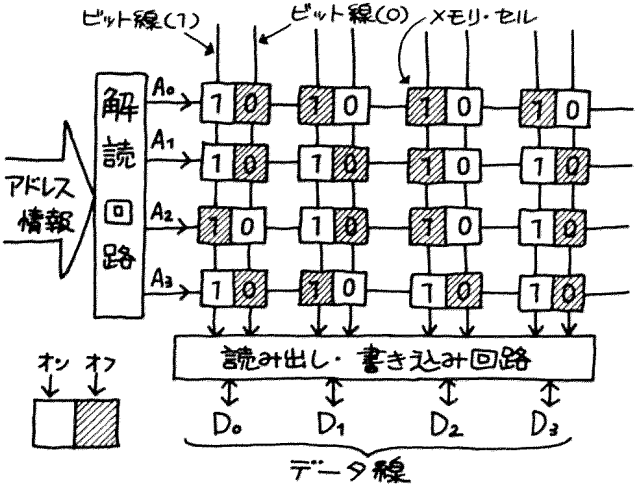


図2－34　RAMの簡単な構成図

です。

②の部分には、上の方から常に電圧がかかっているので、③の部分がデータ線に接続していれば電流がデータ線に流れ、接続していなければ流れません。

したがって、選ばれた番地がA_2なら、D_0＝1、D_1＝0、D_2＝1、D_3＝0となり、A_2番地が記憶している情報は1010であることが読み出せる仕組みです。

このことからわかるように、ROMでは、回路が接続しているか、いないかで、1、0を記憶しているのです。ですから、コンピュータの電源を切っても、記憶内容は失なわれません。

ところが、RAMでは、電源が入っている時は記憶していますが、電源を切ると、すべて失ってしまうのです。

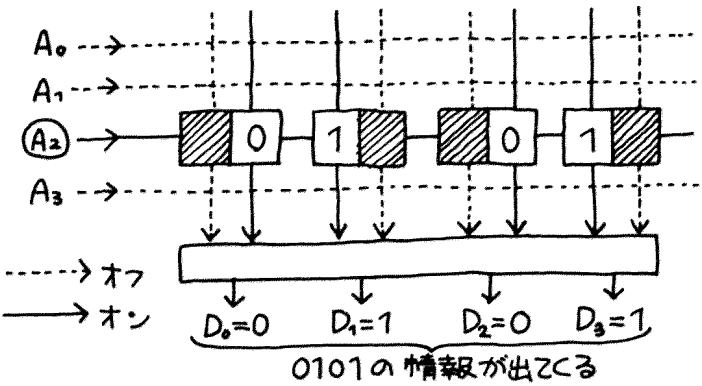


図２－35　図２－34でA₂が選ばれた場合

RAM

ＲＡＭでは、記憶の仕組がＲＯＭよりも複雑になっています。

ＲＡＭにはＬＳＩの種類によって、スタテック式とダイナミック式があります。

(1)　スタテック式

スタテック式は、二本ある線のうち、どちらに電流が流れるかで、1、0を記憶します。一度情報を記憶すれば、次の書き込み命令が来るまで、記憶が保たれています。

図2―34に、その機能の概略を示します。二本あるビット線のうち、どちらに流れるかで、1、0を決定します。ここでは、オンか、オフかではなく、どちらの線にオンかで情報を伝えるのです。

メモリ・セルの内部では、ビット線(1)、(0)

にそれぞれ接続した二つの回路が組合わさっていて、何かを記憶している時は、常にどちらかの回路に電源が流れています。

アドレス情報（読み出し）が入ると、その電流がビット線に移されて記憶情報を伝えます。逆に、書き込みのアドレス情報が来た時は、ビット線(1)、(0)のうち、電流の流れている方に接続している回路に電流が流れ、ビット線がどちらもオフになっても、その回路に電流が流れたままになります。

(2) ダイナミック式

ダイナミック式は、コンデンサが電荷を帯びているか、いないかで、1、0を記憶します。この方式だと、記憶が不安定で放っておくとすぐに記憶を失ってしまうので、千分の2秒ごとに、記憶をふたたび書き込まなければなりません。けれども、構造が簡単なので、限られたスペースに多量の情報を記憶できるという点では、スタテック式よりも優れています。

記憶の読み出し、書き取りでは、コンデンサに電荷が蓄えられていれば、データ線に1の情報が流れ、蓄えられていなければデータ線に0の情報が伝わります（つまりオフ）。また、データ線に1の情報が流れればコンデンサに電荷が蓄えられ、データ線が0であれば、蓄えられません。基本的には、図2—34と似たようなことになります。

以上のようなCPUとメモリ部とが連動して、マイコンの頭脳としての働きをします。通常、マイコン・キットとして売られているものは、このCPUとメモリ部、およびいくつかのインタフェース回路を、一枚の板の上に取り付けるようになっているものです。キ

ーボード入出力装置が付いているものもありますが、このマイコン・キットに入出力装置を取り付けなければ、何の役にも立ちません。

入出力装置

CPU、メモリ部に対して、マイコンのからだともいうべき装置です。I／O装置とか、単に端末とかいうこともあります。

I／Oポート

I／O装置は、マイコン本体と、まったく異なる機械なので、作動する条件がまったく違っています。そこで、I／O装置とマイコン本体（CPUとメモリ部）との間に、両者の条件を一致させてやるものが必要です。

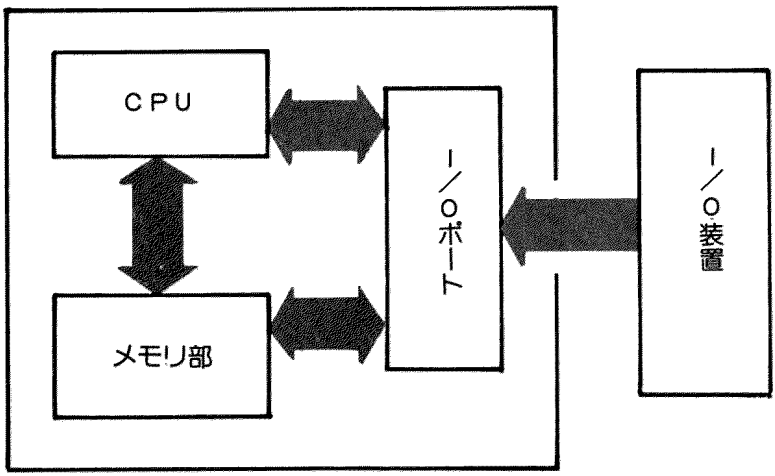
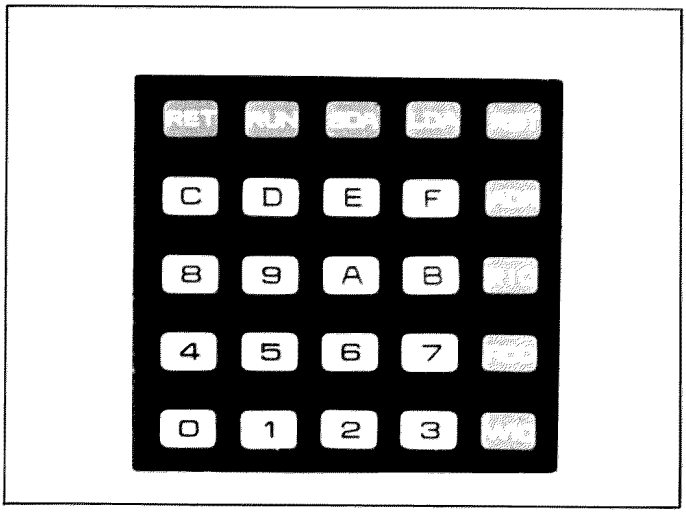


図2－36　I/Oポートとほかの部分との関係
（ポートとは港，発着所の意味）



電卓型キーボード

この役目を果たすＬＳＩのことを、Ｉ／Ｏ（アイオー）ポートと呼んでいます。単に、インタフェースとか、アダプタとかいうこともあります。

インタフェースという言葉は、Ｉ／Ｏポートのほかに、ＣＰＵとメモリとを接続するＬＳＩの呼び名にも使っています。一般に、二つ以上の部分を連動させるためのＩＣを、インタフェースと呼んでいます。

Ｉ／Ｏポートは、ＬＳＩでできたインタフェース回路のことです。いろいろな回路を、組合わせたボード式のインタフェースもあります。

入出力装置の種類

入出力装置には、情報を入力するためのコンソールや、マイコンが仕事をした結果を印刷する印字機、などいろいろあります。

(1) コンソール

命令やデータを入力する装置です。外観がポケット電卓のようなキーボードになっているものと、タイプライタのようなものがあります。

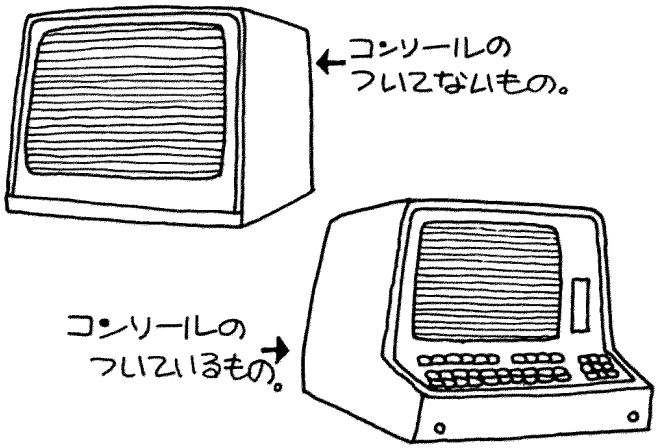
電卓型の方は、やはり電卓のような表示板がついていて、今入力しているデータに押し間違いがないか確かめられるようになっています。また、計算結果も表示されます。

タイプライタ型の方は、ブラウン管や印字機と組合わされたものが多いようです。

(2) オーディオ・カセット

外部メモリとして、オーディオ・カセットを使用することもできます。よく使うプログラムをカセットテープに「録音」しておけば、必要な時にいつでもその内容をRAMに移しかえられます。このような外部メモリに対し、

タイプライタ型キーボード



キャラクタディスプレイ

ROMやRAMを内部メモリといいます。

(3) キャラクタ・ディスプレイ

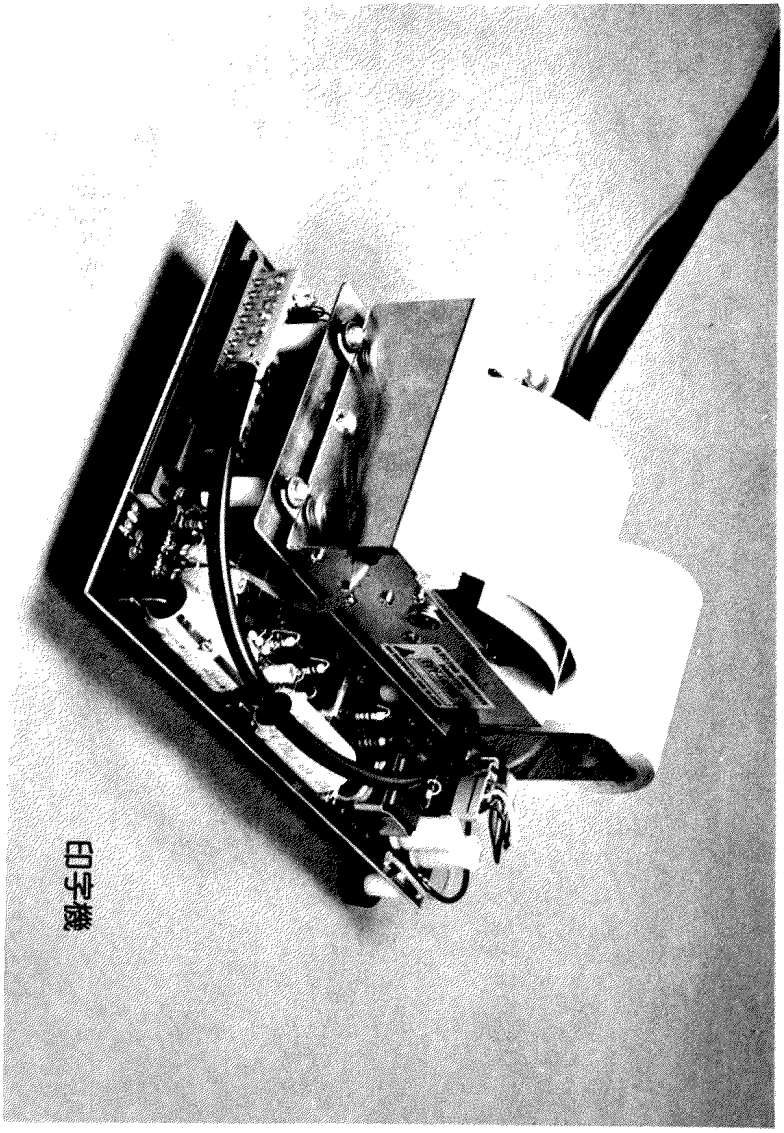
TVブラウン管のことです。家庭にあるテレビでも同じことができますが、コンピュータ用のキャラクタ・ディスプレイの方が、より細かな像を映し出せます。インタフェース次第によって、文字を映すだけでなく、色付きの絵も映し出すことができます。

(4) 印字機

結果をロール状の紙に印字する装置です。小型でスピードの遅いものと、大型でスピードの速いものがあります。

(5) そのほか

紙テープに穴をあけて情報を記録する紙テープ・パンチャーや、紙テープを解読する紙テープ・リーダー、いろいろな入出力装置の複合されたもの、などがあります。


印字機構

マイコンの情報の流れ

マイコンの中をデータや、プログラムなどの情報がどう流れていくかを、図2―37をもとにして説明します。

データ・バス、アドレス・バス、制御バスとは、まえに説明したように、情報を伝える線の集まりです。アドレス・バスはアドレス情報を、制御バスは制御信号を伝えます。データ・バスは、データだけでなくプログラムの情報も伝えます。要するにメモリに入っている情報を伝えたり、逆にメモリを入れる情報を伝えるバスです。

アドレス・バスは、CPUから各部へと一方向にしか情報を伝えませんが、データ・バスはメモリやI／Oポートから出る情報も、また、メモリやI／Oポートに入っていく情報も伝えますから、矢印は両方向に向いています。

ROMに書き込まれているモニタ・プログラム（制御プログラム）は、基本的な制御の命令が書き込まれています。例えば、コンソールのキーを押して情報を入力すると、その情報の取り扱いをどうすればよいか、などについての命令が書き込まれています。

ですから、マイコンを使うたびに入れるプログラムに、書かれている以外のマイコンの動作は、すべてROMの中の制御プログラムによるものです。ここでは、ROMについては、触れませんが、ROMはそういうものです。（図2―38）

1° 入力装置のキーを押してプログラムを入力します。指定されたRAMの番地へ、機械語に翻訳された情報が次々と格納されて行

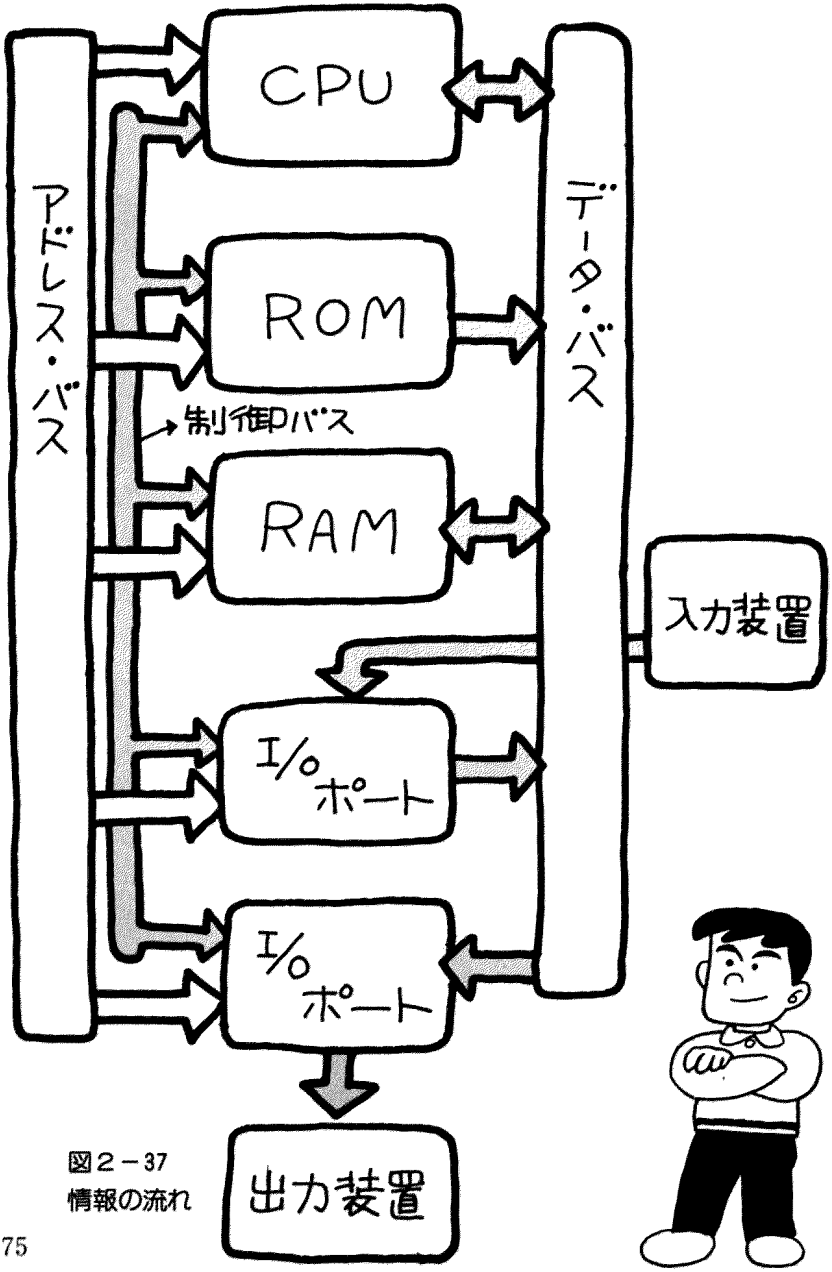


図2－37
情報の流れ

きます。データも、同じように格納されて行きます。（図2―39）

2° すべての情報を入力し終ったら、プログラムの頭から実行するように命令します。CPUは、RAMに格納されている情報を一つ一つレジスタに取り込んで、それにしたがって制御を行います。

ALUは、与えられた仕事をレジスタを活

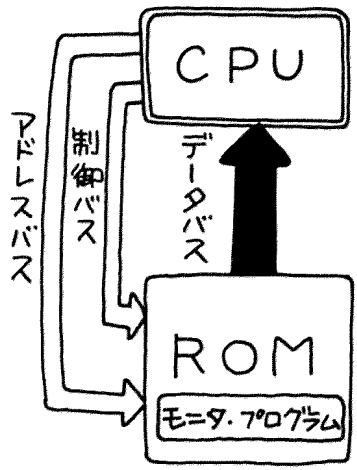


図2－38　CPUとROM

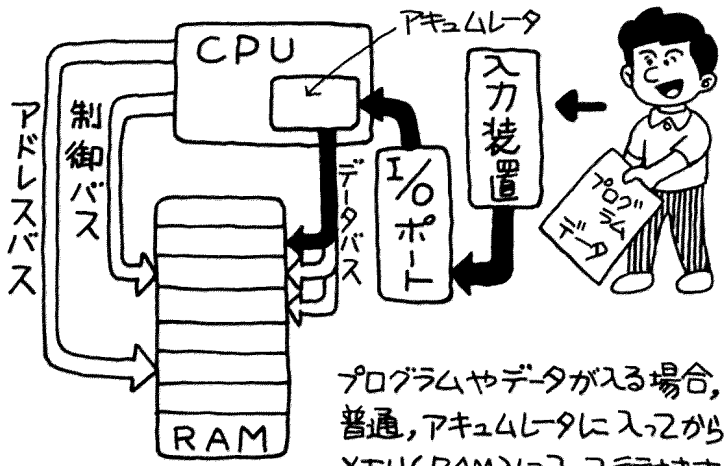


図2－39　1°

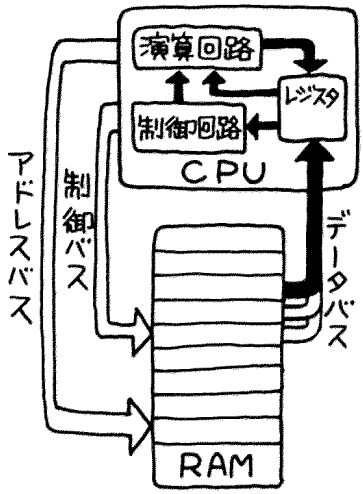


図2－40　2°

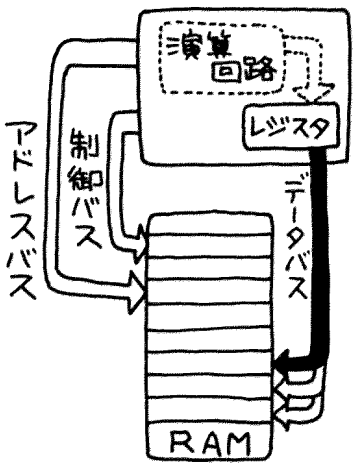


図2－41　3°

用して行います。（図2―40）

3° 実行した結果は一時、レジスタにとどめられたあとで、RAMの指定された番地に移されます。（図2―41）

4° RAMに記憶された結果が、出力装置に送り込まれ、出力装置は、それを具体的な映像や文字にかえます。（図2―42）

以上でマイコンの作業は終りです。制御信

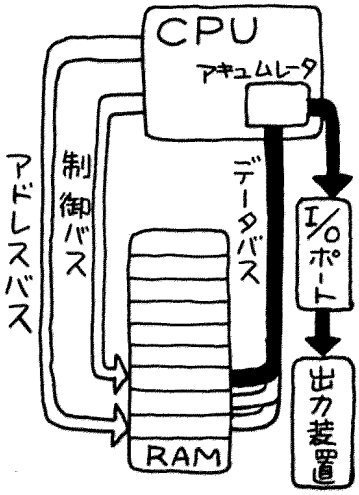


図2－42　4°

号、アドレス信号は、その時その時に作動している箇所を指定し、動作の種類を命令しています。

ハードウェアとは

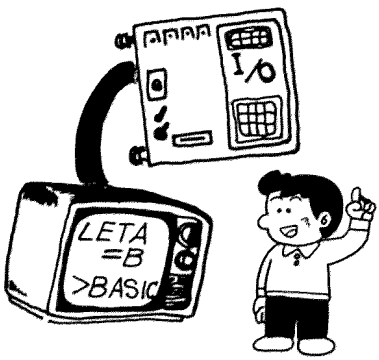
この章に書いてきた事柄（マイコンの仕組み、機能）のことを、ハードウェアといいます。

ハードウェアとは、「金物」の意味で、コンピュータの機械そのものや、その働きのことを指します。

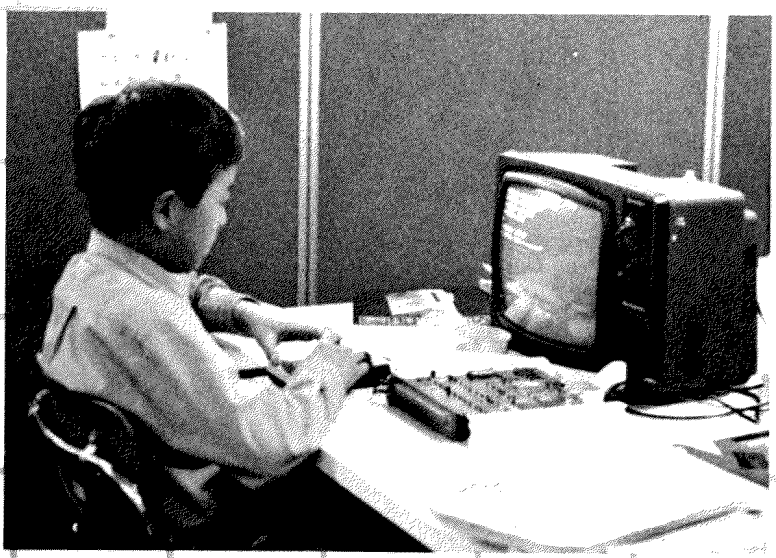
これに対して、プログラムや、コンピュータの使い方などすべてのことを、ソフトウェアといいます。

コンピュータを動かすには、ハードウェアとソフトウェアの両方の知識が必要です。次の章で、マイコンのソフトウェアについて、説明します。

なお、ROMは、プログラム（ソフトウェア）の書き込まれた部品（ハードウェア）なので、ソフトウェアとハードウェアの中間という意味で、ファームウェアということがあります。



★第3章
マイコンと話をする



人間の言葉とコンピュータの言葉

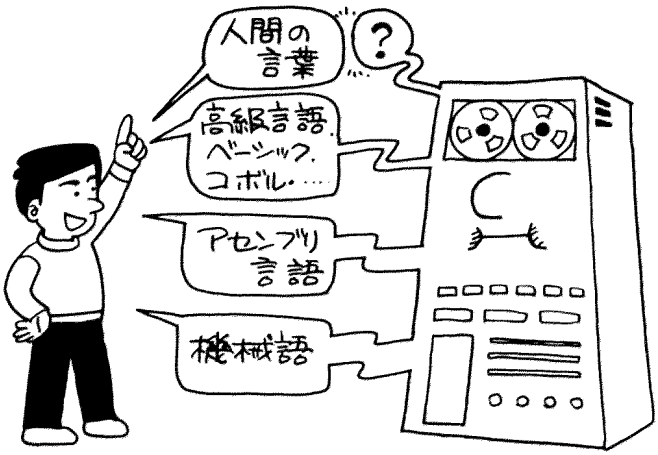
人間は、実は複雑なことを一見単純にやってのけます。これと反対に、コンピュータは一見単純なことを非常に複雑に行います。

よく考えてみると、コンピュータの頭の仕組みや機能などは、かなり複雑で専門家でないとわからないとはいえ、とにかく明確に知ることができます。人間の体や頭脳の仕組みについては、どうでしょうか。医者や心理学者などの専門家は、それらについて明確に知っているでしょうか。答はノーです。実のところ、ほとんど何もわかっていないといっても間違いではありません。

一応は、医学、心理学などの知識体系があり、それを学ぶことによって、人間の体や頭のことを知ることができる、といわれています。けれどもわからないことだらけなのです。その証拠に、いろいろな学説が乱れ飛び、互いに批判しあっていますが、第三者の目からみると、どちらのいい分も五十歩百歩です。例えば、現代医学では治しきれない病気がたくさんあるし、今までさかんに使ってきた薬が、実は大変な毒だったということがいくらでもあります。これは、人間があまりにも複雑にできているからです。

その複雑な人間の使う言葉は、当然複雑です。自分の使う言葉を、ふだんそう複雑には感じませんが、この人間の言葉の仕組みもまた、解明されてはいないほどです。

ちょっとした言葉が、実は多くの情報を伝



人間の言葉とコンピュータの言葉

えているのだということは、まえに述べました。ここでもう一つ例をあげてみましょう。

おとうさんが、「タバコを買ってきてくれ」といって二〇〇円くれた、とします。普通の人なら、「はい」と答えて、おとうさんの望んだ通りにできるでしょう。この単純そうな言葉の中に、どれほどの情報がかくされているか、考えてみましょう。

まず、二〇〇円手渡されたのですから、「二〇〇円の金で、タバコを買ってこい」という意味でしょう。

タバコとは一体何でしょう。タバコという言葉を聞くと、あるイメージが頭に浮かびます。火をつけて煙を吸い込むやつです。しかし、タバコにも、いろいろあります。

葉巻き、紙巻き、パイプタバコ、キセルタバコ、プロ野球の外人選手がよく使う噛(か)みタ

情報とは？

バコ、そのほかにも、臭いを嗅ぐだけの嗅ぎタバコなどもあります。

この中から、あなたは迷わず紙巻きタバコを選んでしまいます。しかし、紙巻きタバコにも、セブンスター、エプソン、ハイライト、ケント、ラッキーストライク、ラーク、など数えきれないほどの種類があります。ここで、あなたは少し考えるかも知れません。

「確か、おとうさんはいつもハイライトを吸っている」

それで、あなたはハイライトだと考えます。ハイライトは今一二〇円ですから、二〇〇円だと、一個しか買えません。ですから、「タバコを買ってこい」という命令は、「タバコにも葉巻き、紙巻きなどいろいろあるが、その中の紙巻き、しかもハイライトという国産銘柄を、一個一二〇円だからこの二〇〇円の金で

一個買って、おつり八〇円もらってきなさい。おつりの八〇円はお駄賃だ。」ということになります。これでやっと、命令の意味がわかりました。しかし、問題はこれだけではありません。「どこで」という命令が欠けています。

一体、「どこで」買えばいいのでしょう。おとうさんは、どこで買えとは一言もいっていません。それでは、どこで買ってもいい筈です。ということは、どこで買うか自分で決めなければなりません。

ここで、あなたは頭の中に家のまわりの地図を思い浮かべ、その中で、一番近いタバコ屋か、自動販売機に買いに行くことを決めて、家を出るでしょう。

これだけのことを、人間はたいてい一瞬にして判断し、行動します。(図3—1)

そういうことができるのは、たくさんの情

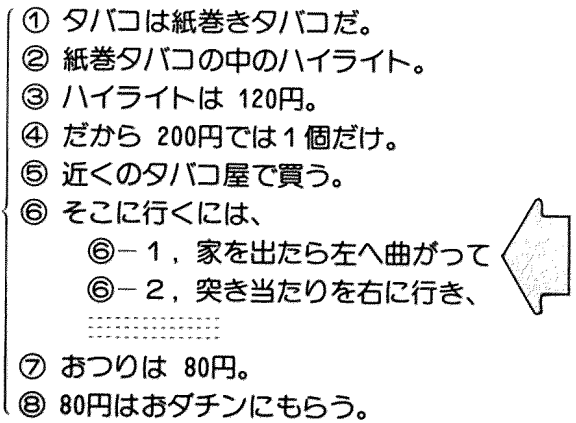


図3－1 情報と判断

報を記憶しているからです。前後の状況、命令を出す相手についての知識、言葉の意味、それらを総合して、命令の意味を判断しているのです。例えば、社長秘書が社長に「タバコを買ってきてくれ」といわれたら、秘書は社長がいつも吸っているタバコ（例えばハマキ）を、すぐ買ってくるでしょう。そういうように、ある命令を判断するには、記憶している情報が必要で、記憶している情報量が多ければ多いほど、命令の形は単純でいいのです。

コンピュータ、特にマイコンは、人間に比べて記憶できる情報はごく少ないものです。マイコンだと、数千語しか記憶できません。そのために、コンピュータに命令を与える時は、微に入り細に渡って命令してやらなければならないのです。

また例えば、日本語を少し話せる外人がしゃべったとしましょう。

「ワタシ、ゼンゼン、スシ、スキデス。トテモ、アジ、ヨロシイデス。」

これは、正しい日本語ではありません。日本語の文法からは、確かにはずれています。厳格に文法を適用するなら、この外人の言葉はまったく無意味だといっていいでしょう。

でも、この外人が、「私は非常に鮨が好きです。鮨はとてもおいしい」といってることくらい、わかります。かといって、日本語に文法がないわけではありません。日本人が、この外人のような言葉使いをすれば変にみえますし、このような言葉使いで、学校の作文を書いたとしたら先生に注意されます。日本語にも、英語にも、フランス語にも、ポリネシア語にも文法があります。しかし、その文法

からはずれていても言葉の意味を理解することができる。これが、人間の言葉のもう一つの大きな特徴です。

それに対してコンピュータの言葉は、厳格な文法を持っていて、それから少しでもはずれると、さっぱり理解できません。

人間の言葉は、複雑なためにあいまいなところがあり、コンピュータの言葉は、逆に単純なために厳密なのです。

コンピュータの言葉

コンピュータの言葉には、多くの種類があります。昔は、コンピュータを作る会社によって、コンピュータの言葉は異なっていました。現在は、どのコンピュータにも共通して使えるプログラミング言語が開発されています。

コンピュータの言葉には、機械語、アセンブリ言語、ベーシック言語、フォートラン、コボル、アルゴル、PL／Iなどがあります。人間の言葉に近いコンピュータの言葉を高級言語ということがあります。ベーシックフォートランなどは高級言語の中の一つです。

機械語

数字ばかりでできている言葉です。コンピュータは、この機械語しか理解できません。CPUやメモリなどの情報交換はすべて、この機械語で行われています。

数字ばかりの言葉だといっても、2進数、つまり、0と1ばかりの数字です。実際には、前にも書いたように、0にオフ、1にオンを

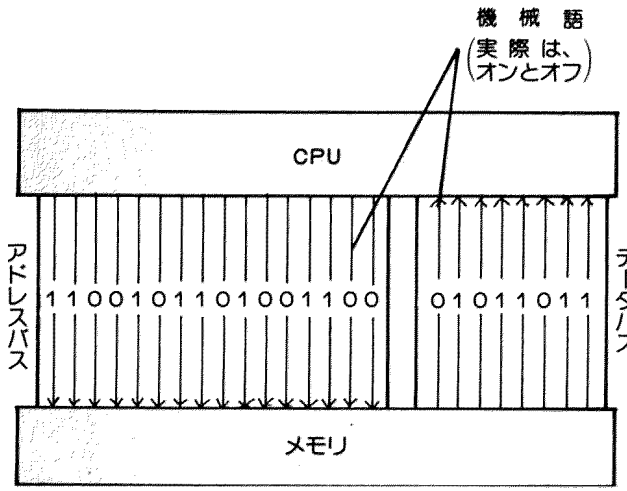


図3－2　機械語は0と1ばかりの言葉

対応させているのですが、人間の方で、それを0、1といっているだけなのです。

ところで、この2進数の並び、例えば8桁の2進数を使って、コンピュータに命令するのは、とても面倒です。何しろ、1と0しかない数字の並びですから、機械語の命令をあらかじめ紙にメモしておく時は、目がチカチカして書き間違えたり、見間違えたりしやすいものです。その命令を入力する時も、（その時は、8ビット型なら8個のスイッチをオンにしたりオフにしたりして入力するのですが）間違えやすいものです。

そこで、長い桁をいくつかに区切って、まとめておくことがあります。

よく行われているのは、2進表示を16進表示にまとめる方法です。

2進数を16進数にまとめるとは、どういう

2進法	10進法	16進法
1	1	1
10	2	2
11	3	3
100	4	4
101	5	5
110	6	6
111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F
10000	16	10
10001	17	11
10010	18	12
10011	19	13
10100	20	14
10101	21	15
10110	22	16
10111	23	17
11000	24	18
11001	25	19
11010	26	1A
11011	27	1B
11100	28	1C

表3－1　2進法、10進法、16進法の比較

ことでしょうか。八桁の2進数を例にとって説明しましょう。

16進法とは、2進法と同じように、16を一区切りにしている数の表わし方です。2進法の一桁の数字に2がなく、10進法の一桁の数字に10がないのと同じく、16進法の一桁の数字も16はありません。

16進数で、使う数字は0から15までです。10、11、12、13、14、15という数字は、10進法の表わし方で二桁になっていますから、16進法では、これを一桁で表わすために新しい数字を作る必要があります。0から9までの数字と区別できるものなら何でもいいのですが、普通、A、B、C、D、E、Fを使っています。これは、それぞれ10、11、12、13、14、15と対応しています。

16進法で、

3F

と書けば、3×16＋15＝63ですから、63のことです。2進法の時と同じように、10進数と区別するために、

$(3F)_{16}$　または　3F　H

などと書くことがあります。

さて、八桁の2進数を16進数に直してみましょう。まず、2進数八桁を半分に割ります。2進数四桁分は、0000～1111、すなわち0から15まで表わせます。これは16進数の一桁で表わせる数と同じです。だから、八桁の2進数なら、四桁ずつが16進数一桁で表わせるので、結局、16進数二桁で表わせるのです。例えば、11010010、01001011を16進数に直してみましょう。

0 1 0 0 1 0 1 1		2進数	1 1 0 1 0 0 1 0		2進数
4	11	10進数	13	2	10進数
↓	↓		↓	↓	
4	B	16進数	D	2	16進数
まとめて書くと			まとめて書くと		
4 B		16進数	D 2		16進数

このように、八桁の2進数なら二桁の16進数、一六桁の2進数なら、四桁の16進数に簡単に直すことができ、しかも、16進数なら数の見分けがつきやすいので、機械語としての数は16進数で表わされることが多いのです。

機械語の命令を16進数で入力する時は、16進キーボードというもので入力します。

マイコン・キットとして売られているものは、そのままでは機械語しか使えないものが多く、機械語は、使用しているマイクロ・プロセッサによって異なっています。

アセンブリ言語

機械語を16進数で表わして見やすくしても、やはり数字では人間にとって扱いにくいことに変わりありません。機械語一つ一つに対応している命令の意味を、数字を見ただけでは思い出しにくいものです。

そこで、命令の意味を簡単なアルファベットの略記号で表わして、機械語一つ一つにそ

の略記号を対応させたものがアセンブリ言語です。機械語としての数字の列を、命令コードあるいは、オペレーション・コードというのですが、一つの命令を表わす命令コードに、一つの英略語を対応させたものが、アセンブリ言語なのです。

インテル8080Aというマイクロ・プロセッサの機械語と、それに対応しているアセンブリ言語で例を示してみましょう。

インテル8080Aの中には、いくつかのレジスタがあり、それぞれに名前がついています。その名前ごとに番地がついていて、3ビットの2進数で表わされています。そのレジスタの中のAレジスタ（番地は111）に、Bレジスタ（番地は000）が記憶している内容を転送せよという命令をみてみます。

これは、命令コードでは、01111000と表わされます。これに対応するアセンブリ言語での命令は、MOV A, Bと表わされます。このMOVという英略語は、MOVE（ムーブ）という英単語からきています。ムーブとは、動かす、動くという意味の言葉です。

01111000を見ただけでは何のことかわからなくても、MOV A, Bだと、Bレジスタの内容を、Aレジスタに「ムーブ（動かす）」するのだな、ということがすぐにわかるわけです。

入力用のキーボードで、アセンブリ言語を入力すると、メモリに入っているプログラム（アセンブリ言語を機械語に翻訳するためのプログラム）によって、アセンブリ言語は機械語となって、マイコンが命令を受け取る仕組みになっているのです。

アセンブリ言語でプログラムを書き、それを入力する場合、そのプログラムをソース・

プログラムといいます。ソースとはみなもとの意味です。ソース・プログラムを、機械語に翻訳するためのプログラムのことをアセンブラといい、アセンブラによって翻訳された機械語のプログラムをオブジェクト・プログラムといいます。オブジェクトは目的、対象の意味です。

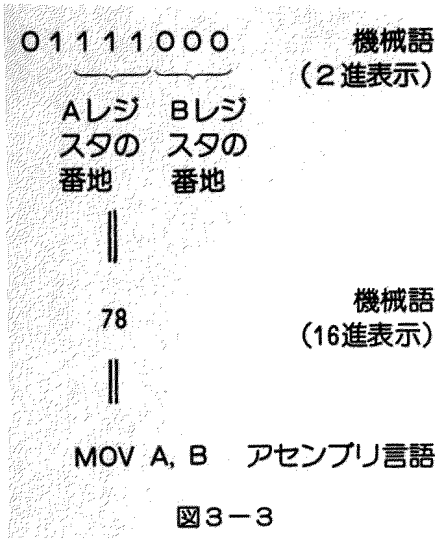


図3－3

アセンブリ言語は、機械語よりは使いやすいものですが、機械語と一対一に対応しているので、結局は機械語とあまり変りありません。ただ、数字ばかりの機械語に、意味がすぐわかるようなアルファベットを貼りつけただけのものです。機械語というのは、あとでもう少し詳しく説明しますが、コンピュータの中の情報の流れだけを命令したものなのです。例えば、3と8を足せ、という命令でなしに、3と8のデータを何々レジスタに入れ、それを加算機に通して、出てきた情報をメモリのどこどこに入れて、それをどこどこの出力装置に送れ、という命令なのです。

ですから、機械語は数字だけでできているという点と、命令の仕方が非常に複雑だという点とで不便なのです。数字だけでできているという不便さを解消したのが、アセンブリ

言語なのですが、命令が複雑だという点は、アセンブリ言語でも機械語でも同じように不便なのです。そこで、さらにその点も解消しようという言語が作られています。

高級言語

なぜ、人間の言葉に近い言語を高級言語というかというと、人間の言葉は単純な形をとっていますが、内容は非常に複雑で、機械がそれを受け入れるためには、複雑な手順が必要だからです。

高級言語には、ベーシック、フォートラン、コボル、アルゴル、PL／Iなどがあり、それぞれ、ほとんど英語を基本にして作られたものです。マイコンでよく使われる高級言語は、ベーシックです。ほかのものは、あまりマイコンでは使いません。

(1) ベーシック

ベーシックは、もともとミニコンなどで使われていた言語ですが、ベーシックを機械語に翻訳するプログラムが、ほかの高級言語よりも比較的簡単なので、マイコンにも使われるようになりました。マイコンで普通使われているベーシックは、tiny－BASIC（タイニー・ベーシック、「小さいベーシック」の意味）といって、さらに簡単にされたものです。そのかわり、タイニー・ベーシックは、もとのベーシックに比べて限られた命令しかできません。とはいえ、遊ぶには不自由しないようです。ベーシックは、ほかの高級言語と同じく、簡単な英語と数式でできています。例えば、LET C＝A＋Bと書けば、AとBを足して、その答えをCと置きなさいという命令

```
 10  INPUT  N,I
 20  LET  A=N          (タイニー・ベーシックによる)
 30  GOSUB  120
 40  LET  C=F
 50  LET  A=N-I
 60  GOSUB  120
 70  LET  C=C/F
 80  LET  A=I
 90  GOSUB  120
100  LET  C=C/F
110  GOTO  190
120  LET  F=1
130  LET  B=1
140  IF  A<B  THEN  180
150  LET  F=F*B
160  LET  B=B+1
170  GOTO  140
180  RETURN
190  PRINT  "COMBINATION" ;N;  "," ;I; "=" ;C
```

図3－4　ベーシックによるプログラムの例

になります。LETは、「させる」という意味の動詞です。このように、ベーシックは、日常の言葉とそんなに変らない単純な命令でませることができます。

何度も書きましたが、見かけが単純なほど、内容はコンピュータにとって複雑で、ベーシックによるプログラムを機械語に直すためには、複雑な手順が必要になります。

ベーシック言語を解釈し、実行するためのプログラムを、インタープリタといいます。アセンブリ言語の時と同じようにメモリにそのプログラムを入れておかないとベーシック言語を使ってコンピュータに命令することはできません。

```
      REAL TAKASA, TYOHEN, TEIHEN, HIROSA
      READ(5, 101) TAKASA, TYOHEN, TEIHEN
101   FORMAT(F6.2, F7.2, F8.2)
      HIROSA=(TYOHEN+TEIHEN)*TAKASA/2
      WRITE(6, 102)TAKASA, TYOHEN, TEIHEN, HIROSA
102   FORMAT(F6.2, F7.2, F8.2, F15.5)
      STOP
      END
```

図3－5　フォートランのプログラム例

標準ベーシックのインタープリタには、八千バイト（普通、八キロバイトといいます）もの記憶容量を持ったメモリが必要とされます。マイコンで使う、タイニー・ベーシックでも二キロバイトのメモリが必要です。これは、人間がより難しい言葉を理解するためには、より多くの知識がなければならないのと同じことです。

(2)　アルゴルとフォートラン

アルゴルとフォートランは、科学計算用の高級言語です。ベーシックも、もともとは科学計算用の高級言語です。

科学計算というのは、科学で使う計算のことで、高等数学を使って難しい計算を行ったりします。この二つとも、ベーシックと同じように簡単な英語と数式でできています。

(3) コボル

フォートランなどは、科学計算だけを取り扱うものだったので、事務などでのデータ処理（例えば、模試成績のデータ処理や預金額などのデータ処理）には向いていないものでした。そこで開発されたのがコボルです。

コボルは、ほとんど英語の文章に近いもので、数式を使わずに計算の命令ができるものです。

(4) PL／I

科学技術計算にも、事務処理にも使える、より広範囲な用途を持った言語です。

フォートラン、PL／I、などの高級言語を機械語に翻訳するためのプログラムをコンパイラといいます。

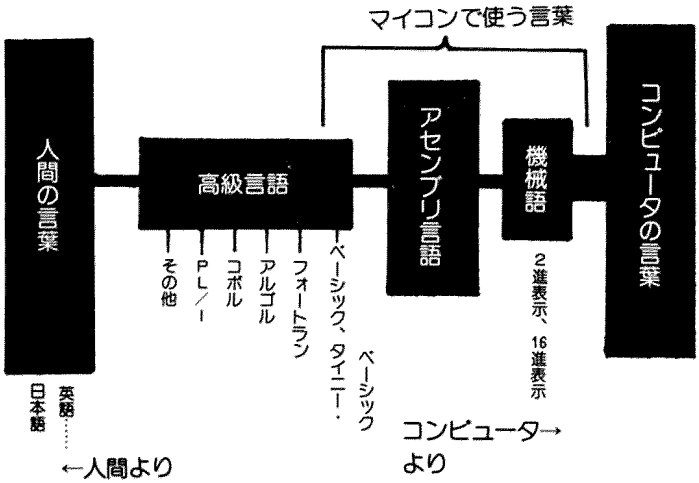


図3－6　人間の言葉とコンピュータの言葉をつなぐ言葉

プログラムって何だろう

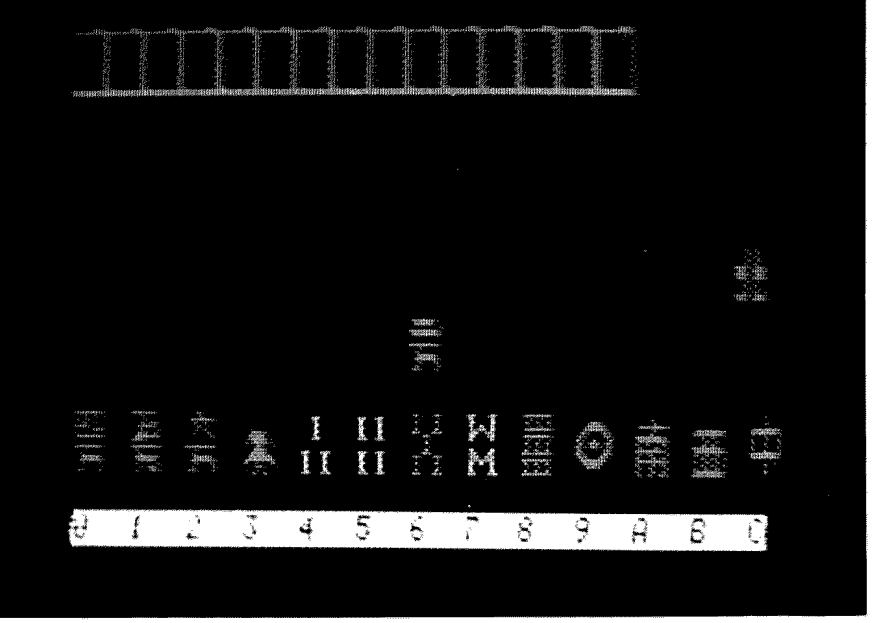
プログラムは、今まで書いてきたコンピュータ用の言葉を使って作ります。マイコンでは、機械語、アセンブリ言語、ベーシックを使ってプログラムを作っています。PL／Iを簡単にしたPL／Mという言葉も使われますが、おもなものは、まえにあげた三つです。ではプログラムとは一体何でしょうか。

プログラムという言葉は、いろいろなところで使われています。テレビ番組のプログラム、コンサートのプログラム、入学式のプログラムなどがあります。これらに共通することは、いずれも行われることを順々にしるしてあることです。

コンピュータのプログラムも、コンピュータにさせたい仕事を順番にしるします。少しでも順番を間違えると、コンピュータは混乱して、答を出してくれないか、とんでもない答えを出したりします。

コンピュータでのプログラムとは、命令を順々に並べたもの、というわけです。順々に並んだ命令を、コンピュータは順々に実行して行き、最後に答えを出します。プログラムの種類によっては、単純に答えを出すのではなくて、その時、その時の機械の動きを制御するものもありますが、やはりそこでも命令は順々に実行されて、その結果として、機械を制御する信号を発生するのです。

命令の順番も、一つの情報です。ある命令が、ほかの命令のあとにくるか、まえにくるか、始めから数えて何番目の命令かによって、



マイコンによるマージャン。プログラムを組まないとできない

行うことが違ってきます。

それは、プログラムを構成している命令が、まえに出てきた命令と深く関わりあってできているからです。例えば、フォートランなどで使う数式の中で、

C＝C＋1

というのがあります。普段使う数式では、このような式は、間違いということになりますが、ここでは等号の左のCと右のCとは違うものです。右のCは、この文が出てくるまでの、命令で使われていたCの値を意味し、左のCは、そのCの値に1を加えたものを、意味しています。この文のあとの命令では、Cという文字が出てくると、左のCとして扱うようになります。

ですから、まえの命令で、Cが5であれば、この文のあとからはCは6になり、まえの命

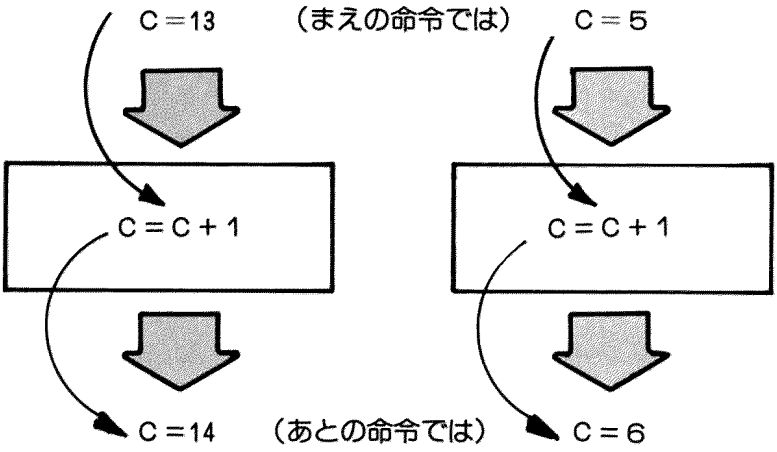


図3－8　まえとあとではCの値がちがう

令でCが13であれば、この文のあとからはCは14になるわけです。（図3―8）

C＝C＋1

という文を実行するまえとあとでCの値が異なるのですから、順番を勝手に入れかえれば答えが違ってくるのは当然です。

プログラムを作るには

プログラムを作ることをプログラミングといいます。

プログラミングするには、まずコンピュータにさせたいことを厳密に決めなければなりません。どういうことをさせるか、はっきりしないと、細かい命令を整然と並べたプログラムなどできはしないのです。

TVゲームのプログラムを作ろうとする時

には、その細かいルールなどをあらかじめ決めなければなりません。

ピンポンゲームで、ラケットを表わす長方形の真中に球がくれば素直にはねかえるけれど、真中より右の方に球がきた時には、右よりにはねかえるようにしようとか、受けそこなって球がうしろにそれたら、その球を打った相手に一点入るとか、どちらかが先に15点取ったらゲームオーバーにするとか、いろいろ決めなければ、そのプログラムはできないのです。

させることが決まれば、次に大体の命令の順番を決めて、図に書きます。この図を流れ図、またはフロー・チャートといいます。

このフロー・チャートにそって、どれか一つの言葉を使って、プログラミングして行きます。

フロー・チャート（流れ図）

フロー・チャートを製作するにあたって、使う記号が定められています。マイコンのプログラミング用には、おもに六つの記号を使います。

(1)ターミナル

ターミナルという言葉は「端」という意味です。一つの流れ図の始めと終りを表わす記

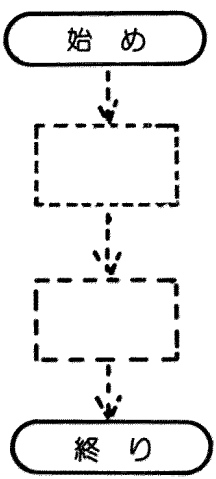


図3－9　ターミナル記号

号です。

(2)判断

ある質問に対して、その答えによって次に実行すべき箇所を指し示すものです。例えばCという文字で表わされる数値が、15以下なら、Aという箇所に飛び、16以上ならすぐ次のBという箇所に行くとすれば、判断を表わすひし形の記号の中に、

C≦15

と書き、YESならAへ、NOなら（この場合、16以上ということです）Bへ行くように線を引くのです。

あるいは図3—11のように書くこともあります。とにかく、判断の記号の中に書かれた

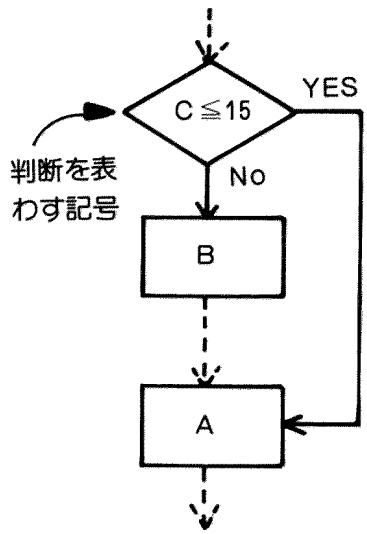


図3－10　判断の記号(1)

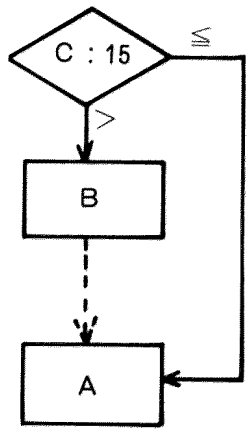


図3－11　判断の記号(2)

ものを判断し、その結果によって進むところを指示するものなのです。

(3)各種処理

データの処理を表わします。例えば、Cの数値に1を足し、その答えを再びCとして取り扱え、などの処理です。

これは、機械語でプログラミングする時と、高級言語でプログラミングする時とで少し違います。高級言語なら、そのまま人間の言葉のように書いておけば充分ですが、機械語だと、「足す」ということをデータの流れで表現しなければならないので、もっと複雑になります。

Bレジスタの内容をAレジスタに入れて、それと、Cレジスタの内容とを加算器に入れて、その結果を……というように。

大体、大まかなフローチャートを書く時は、人間の言葉のように書けばいいのですが、プログラムと一対一に対応するようなフローチャート（これを詳細フローチャートといいます）を書く場合は、使う言葉によって複雑さが異なります。

高級言語でプログラミングする場合でも、処理機能を簡単には表わせない場合があります。その場合は、次の記号と、もう一つ別の流れ図とで表わします。

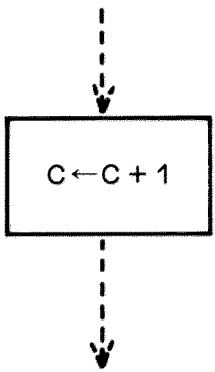


図3－12　各種処理

(4)定義済み処理

別の流れ図で定められた処理手順を表わします。たいてい、サブルーチンを表わすことが多いようです。

サブルーチンとは、繰り返し使われる複雑な手順を別のプログラムで定義したものです。

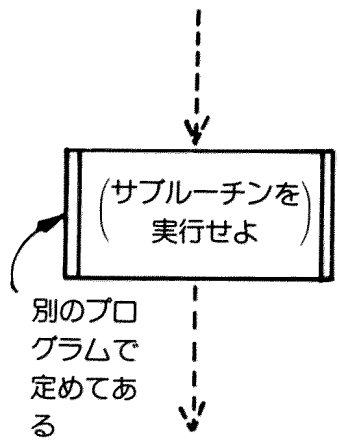


図3－13　定義済み処理

(5)コネクタ（結合子）

フローチャートの出口や入口を表わします。例えば、図3－13のように書けば、上のコネクタから、下のコネクタに流れて行くことを表わします。

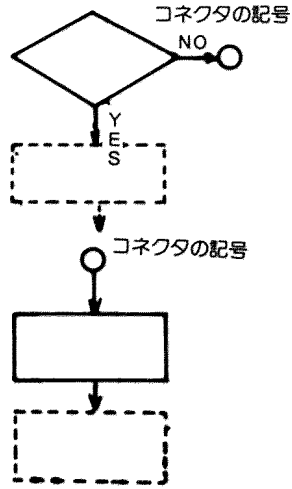


図3－14　コネクタ

(6)流れ線

前の五つの記号を結び、流れて行く方向を矢印で表わした線です。

これらの記号を使って実際にフローチャートを作ってみましょう。

10個のカップラーメンがあるとして、それに次々と熱湯を注げという命令を少し細かく

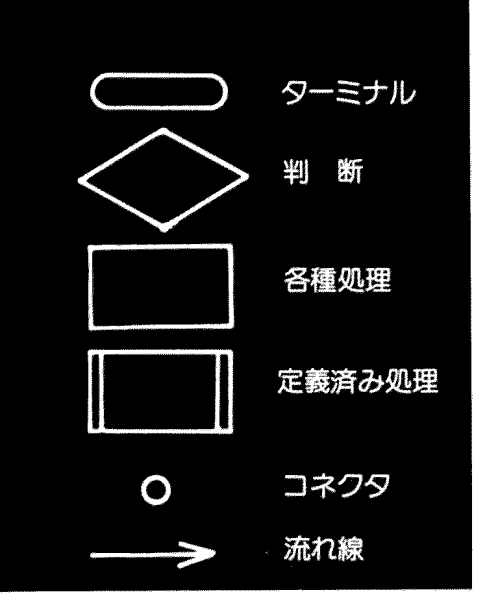


表3－2　フローチャートの記号のまとめ

分析すると、

(1) カップラーメンを一個取る
(2) フタを開ける
(3) 熱湯を注ぐ
(4) フタを閉じる
(5) 三分間待つ
(6) 食べる人に渡す
(7) 次のカップラーメンを取る
(8) (2)～(6)の手順を繰り返す
(9) 10個に熱湯を注げば終り

の九つに分けることができます。ここで、(2)～(6)の手順がサブルーチンになるわけです。

これをフローチャートに書くと図3－15のようになります。

この仕事を、マイコンを組込んだ機械に行わせるとすれば、もっときめ細かな命令を組立てなければならないことはいうまでもありません。

主体となるプログラムを主プログラム、サブルーチンなどの付属するプログラムを副プログラムといいます。図3－15の副プログラムのターミナルに書いてある「リターン」は、主プログラムの同じ箇所にもどって、次の処理に進むことを表わしています。

実際に、マイコン用のプログラミングをす

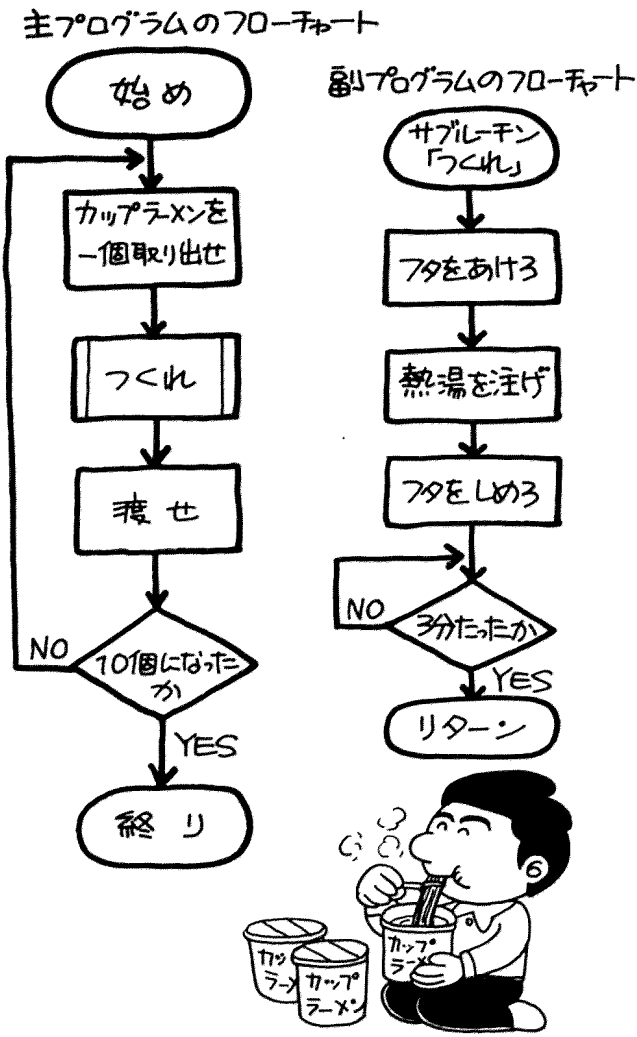


図3－15　カップラーメンのフローチャート

るためにフローチャートを作る場合は、もっと複雑で長いフローチャートを作ることになります。それが機械語かアセンブリ言語によるプログラムであれば、ちょっとしたゲームのプログラムでもノート数ページ分のフローチャートを書かなければならないこともありますし、各記号の結び付き方も複雑です。そういう場合、コネクタを使えば分割して書くことができます。もちろん、どのコネクタとどのコネクタがつながるかを明らかにするために、コネクタの記号の中に、aやbなど記号を入れて区別するといいでしょう。

プログラミング

フローチャートを書き、どれか一つの言葉で命令を書いて行くこと全体をプログラミングといいます。フローチャートにそって、実際のコンピュータ用言語で命令をしるして行くことを特にコーディングということもあります。

(1)アセンブリ言語による プログラミング

機械語とアセンブリ言語とは一対一に対応しているので、機械語のプログラムを作る場合でも、一応アセンブリ言語でプログラミングしておいて、あとで対応表を見ながら機械語におきかえることにすれば楽になります。

機械語はマイコンによって異なるので、当然アセンブリ言語もマイコンの種類によって違いがあります。これは、命令語が各マイコンの内部構造と密接に関係して成り立っているからです。けれども、基本的には似たようなものです。

アセンブリ言語、機械語によって、どのよ

うにプログラムが作られるか、ここではインテル８０８０Ａというマイクロ・プロセッサの機械語、アセンブリ言語を例にとって、簡単なプログラム例を説明して行きましょう。

例１　二つの数の足し算

38＋92の計算をしてみましょう。

まず、フローチャートを書かねばなりませんが、そのまえに８０８０Ａの内部構造を知らねばなりません。

Ａレジスタは、アキュムレータのことです。演算をする時は、一方のデータをここに入れなければなりません。また、演算の結果は、ここに入ります。

Ｂレジスタ、Ｃレジスタ、Ｄレジスタ、Ｅレジスタ、Ｈレジスタ、Ｌレジスタは、汎用レジスタで、必要に応じてデータを入れたり出したりします。それぞれ、単独でも使えますが、ＢとＣ、ＤとＥ、ＨとＬをペアで使うこともあります。

それぞれ、一つのレジスタは８ビット収容できるようになっています。

他にも、いろいろな働きをするレジスタがありますが、ここでは必要ありません。（図3―16）

データを転送するための命令の一つに、まえにも出てきた MOV があります。

MOV A, B

と書けば、Ｂレジスタの内容をＡレジスタに移せという命令です。

また、

MOV A, M

と書けば、メモリの、指定された番地の内容を、Ａレジスタに移せという命令です。

指定された番地とは、Ｈ・Ｌレジスタのペ

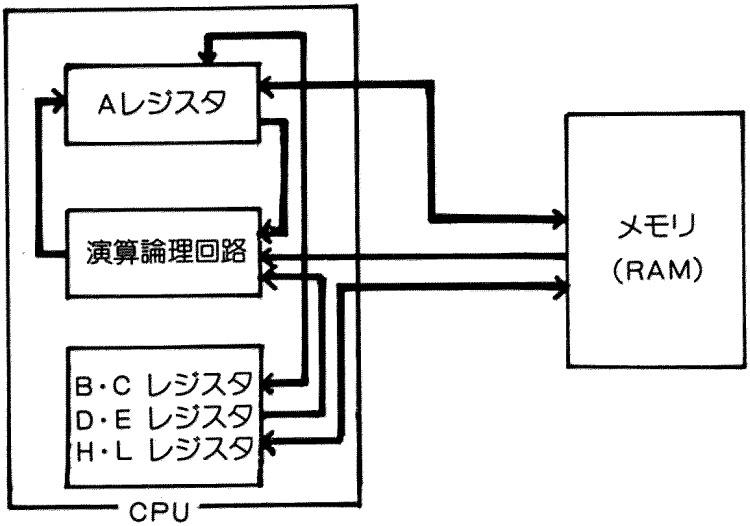


図3－16　CPU内のレジスタ(一部)

アによって指定された番地です。あらかじめメモリから移したいデータの入っている番地を、H・Lレジスタに入れておくのです。

番地（アドレス）は16ビットなので、番地情報を入れるのに、8ビットのレジスタが二つ必要なのです。

図3―17では、16進法で表わしているので、8ビットの情報は、16進数八桁で表わされ、16ビットの情報は、四桁で表わせることになります。

レジスタは、8ビットのデータを一つしか持てないので、番地を指定する必要はありません。

MOVの後にくる、AやBやMなどの並ぶ位置をひっくり返すと、逆の命令になります。

MOV B, A

ならば、Aレジスタの内容をBレジスタに

数字は全て16進数、アドレスは16ビットだから４桁の16進数になり、データは８ビットだから２桁になる。

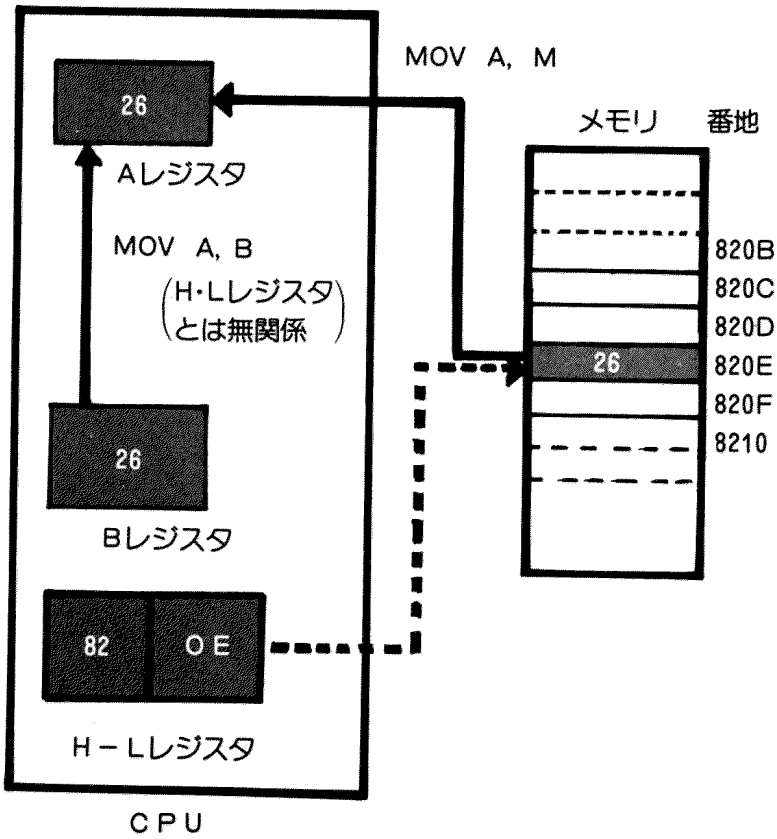


図３－17 MOV A, MとMOV A, B

移せという命令になります。（図3―18）

データの転送を命令する語は他にもありますが、ここではMOVだけを使います。

さて、それではH・Lレジスタに番地情報を入れる命令はどうでしょうか。

LXI H, 820EH

と書けば、H・Lレジスタに、16進数の820Eを入れなさいという意味になります。

最初のHは、LXIの後だと、これだけでH・Lのペアを意味し、820EHのHは、820Eが16進数であることを表わしていま

図3－18

す。

この命令によって、H・Lレジスタに番地情報が入り、MOV A, Mの命令を与えると、その番地のデータがAレジスタに入ることになります。

さて、もう一つ必要なのは足し算の命令です。

ADD B

と書けば、Bレジスタの内容とAレジスタの内容を足せという意味になります。足し算する時は、必ずAレジスタが加わるので、Aレジスタのことはしるす必要がないのです。

ADDとは、英語の「加える」という単語です。

この命令を実行すると、今までAレジスタにあったデータは消えて、かわりにこの足し算の答えがAレジスタに入ります。

これで、フローチャートを書くことができます。

例1の38は、8300H番地、92は8301H番地、答えは8302H番地に入れることにします。番地は全て16進数、数字の後のHは、16進数を表わすHです。

① H・Lレジスタに8300Hを入れる。
② Aレジスタに、H・Lレジスタの指定するメモリの番地の内容を移す。
③ H・Lレジスタに8301Hを入れる。
④ Bレジスタに、H・Lレジスタの指定するメモリの番地の内容を移す。
⑤ AレジスタとBレジスタの内容を足し算する。
⑥ H・Lレジスタに8302Hを入れる。
⑦ Aレジスタの内容を、H・Lレジスタの指定するメモリの番地に移す。

この手順をフローチャートに書くと、図3—19のようになります。

図で、Mは、H・Lレジスタの指定するメモリの番地を意味します。(M)や(B)は、それぞれ、Mの持つ情報、Bレジスタの持つ情報を意味しています。

たかが足すだけなのに、これだけ複雑な手順を踏まねばならないのです。もっとも、使う命令語を変えれば、もっと短くなりますが、大してかわりません。機械語やアセンブリ言語でプログラムを書くということが、どんなに大変か、これでわかるでしょう。

このフローチャートをもとに、アセンブリ言語のプログラムを書くと、表3—3のようになります。プログラム情報は8200H番地から入れることにし、データ情報は8300H番地から入れることにします。

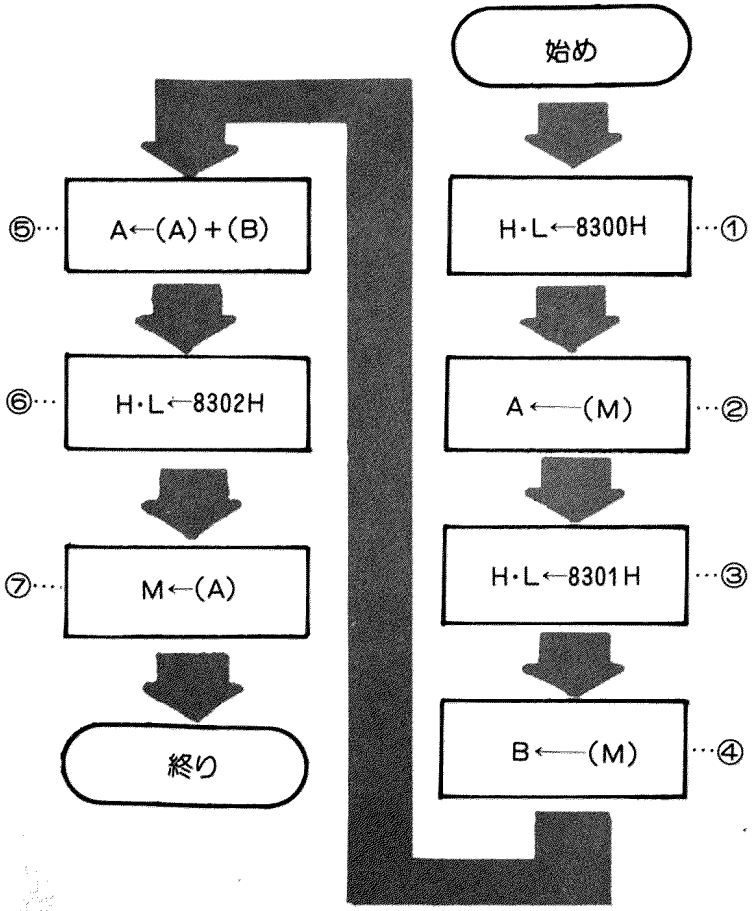


図３－19　足し算のフローチャート

オペレーション欄に書いてある命令語をニーモニックといいます。オペランド欄には、対象となるレジスタやメモリ、番地などを書きます。コメント欄は、書いても書かなくてもいいところですが、命令語の意味を自分のために書きとめておく欄です。ですから、わかっていれば、また、あとでこのプログラムを見て何のプログラムかわかるのなら、何も書かなくていいのです。

HLTは、CPUを停止させる命令です。この命令では、メモリやレジスタの内容は変わりません。

ORG、DB、ENDは、疑似命令といって、ニーモニックではありません。これは、CPUに対する命令ではなく、アセンブラに対する命令で、機械語の命令にはありません。

1　ORG　8200H

8200H番地から、プログラム情報を入れて行くという疑似命令です。機械語では、直接番地を指定しながらプログラムを入れて行きます。（表3―3の左図）

プログラムとデータを入力した場合、アセンブラによって機械語に直され、メモリに格納されます。それで、メモリのどこに入れるかをあらかじめ決めなければなりません。

プログラム情報を入れるメモリのスペースを、プログラム・エリア、データ情報を入れるメモリのスペースを、データ・エリア、といいます。普通、メモリ・マップを書いて、あらかじめ、プログラム・エリア、データ・エリアを決めておきます。（図3―20）

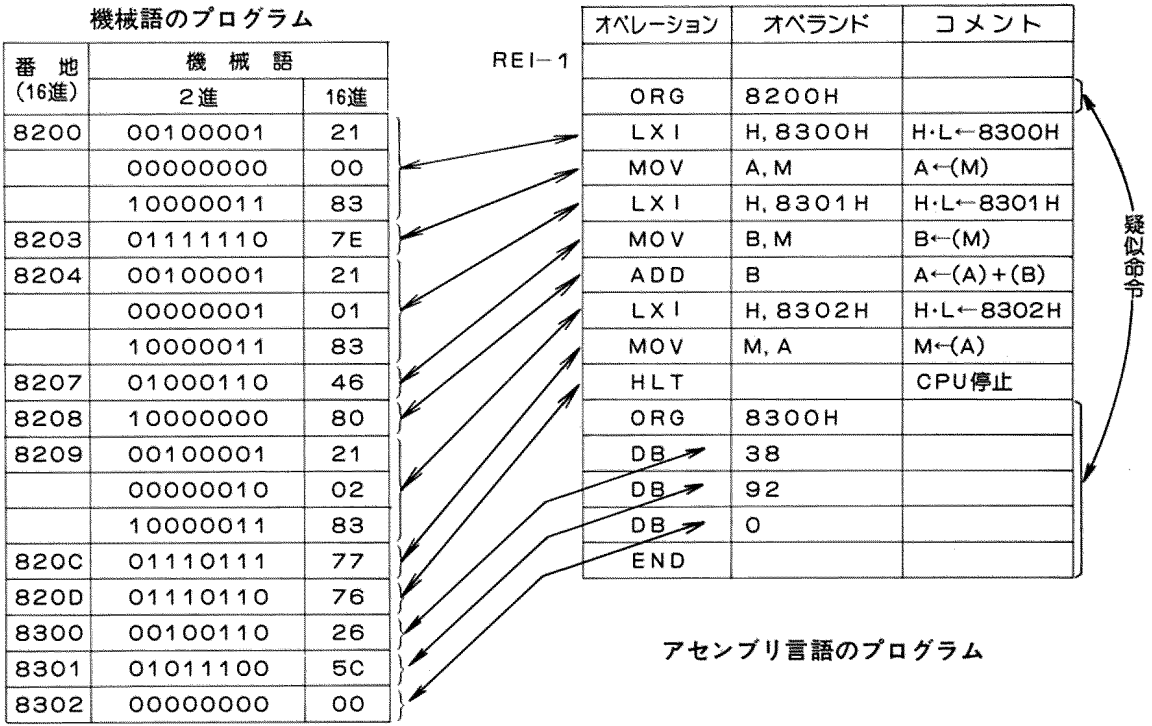


表３－３　アセンブリ言語のプログラム（例１）と機械語の対照

メモリ・マップの中に、情報を書き入れる時は、図のように16進数で入れます。

2　LXI H, 8300H

最初のHは、H・Lレジスタのペアを表わすH、最後のHは16進数を表わすH。

この命令は、H・Lレジスタに、8300Hというアドレス情報を入れよ、という意味です。これによって、次の命令に出てくるMの具体的な番地を指定します。

表3—3を見ると、この一行の命令に対応する機械語の命令は、三行になっています。すなわち、ここの命令に機械語では3バイト使っています。このような命令を3バイト命令といいます。機械語で2バイトになる命令は、2バイト命令、1バイトになる命令は、1バイト命令といいます。

図3—21を見て下さい。8300という情報を、機械語で入れる時は、下位二桁から入れて行きます。メモリにもそのように入ります。

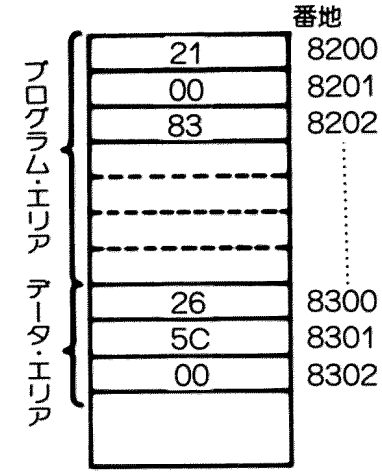


図3—20　メモリ・マップ

3　MOV A, M

H・Lレジスタで指定されたメモリの番地の内容を、Aレジスタに入れる命令です。

この場合、そのデータは16進の26です。

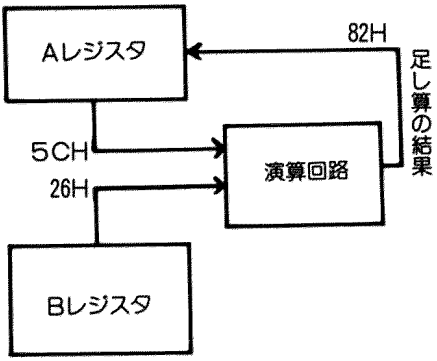


図3－22　ADD　B

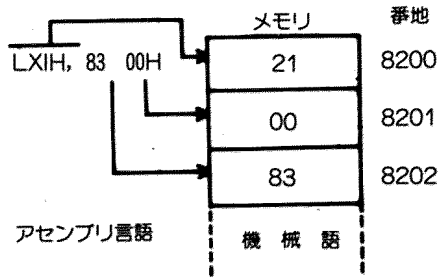


図3－21　3バイト命令

4　LXI　H, 8301H

H・Lレジスタに8301を入れよという命令。

5　MOV　B, M

8301番地の内容をBレジスタに入れる命令。16進の5C。

6　ADD　B

Aレジスタの内容とBレジスタの内容とを足し算する命令。結果は、Aレジスタに入ります。（図3—22）

7　LXI　H, 8302H

H・Lレジスタに8302を入れます。

8　MOV　M, A

Aレジスタの内容（足し算の答え）を、メモリの8302番地に入れる命令。こうして、38（16進の26）と、92（16進の5C）とを足した結果、130（16進の82）が、

メモリの8302番地に入ります。

9　HLT

CPU停止。これでプログラムは終りです。

10　ORG 8300H

8300番地からデータを入れて行くという疑似命令。

11　DB 38

38は10進数です。数字のデータは、そのままだと、10進数として受けとられます。もちろん、アセンブラで2進数に変換されます。38は足し算する一方の数です。

12　DB 92

92は、もう一方の数です。

13　DB 0

ここに、二数を足した答えが入ります。一応0を入れておいて、番地を確保します。

14　END

アセンブラに、プログラムはすべて終ったことを示す疑似命令です。

このプログラムを、機械語で入力する時は、表3—3の左のように、まず番地情報を入れて、その番地に命令が入ることを示してから、命令を一つ入れ、次の命令を入れる時は、また番地情報を入れてからにします。

次に、マイコンに、大小の比較をさせるプログラムを作ってみましょう。

例2　38と92のどちらが大きいか

前の例で書いたフローチャートは、詳細フローチャート（そのままプログラムに移せばよいもの）でしたが、今度は、あらいフローチャートを書いてみましょう。前と同じく、38はAレジスタへ、92はBレジスタに入れる

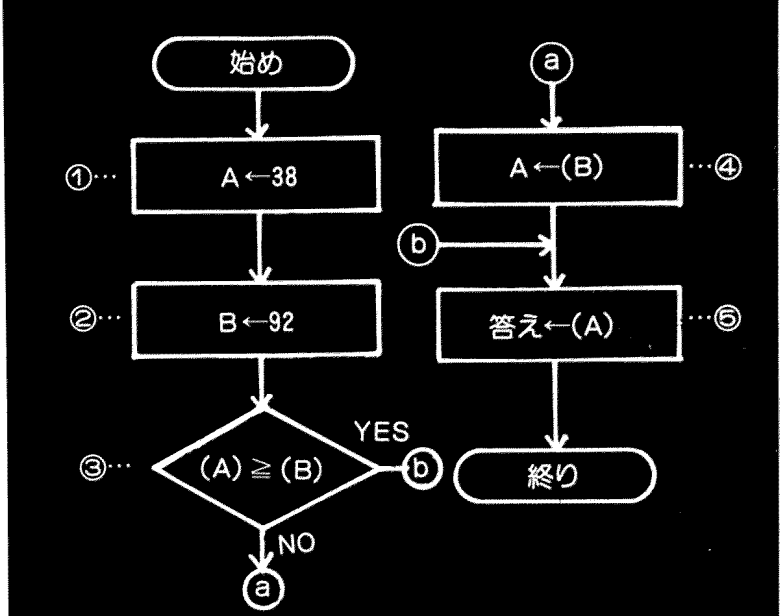


図3－23　例2のあらいフローチャート

ことにします。（図3―23）

38と92を比較し判断するところでは、二つの命令が必要です。

それを説明する前に、フローチャートの説明をしましょう。

① Aレジスタに38（10進数）を入れる。

もちろん、この命令は例1と同じように、二つの命令語で行います。

② Bレジスタに92を入れる。

③ Aレジスタの内容（38）と、Bレジスタの内容（92）とを比較して、(A)が(B)より大きいかまたは(A)が(B)と等しいかを判断し、どちらかならYES、小さいならNO

③―1　YESなら、⑤に飛ぶ。

③―2　NOなら、そのまま次の処理を行う。

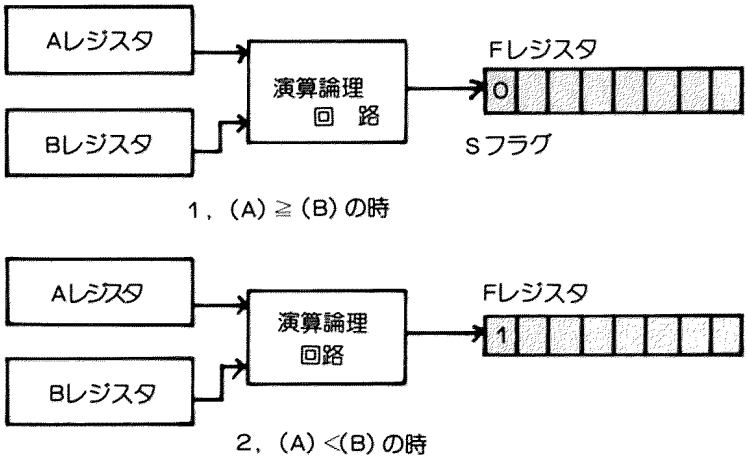


図3－24　Sフラグの状態

④　Bレジスタの内容をAレジスタに入れる。

⑤　Aレジスタの内容を答えとして、メモリに入れる。

③の判断のところで、二種類の命令が必要です。大小を比較する命令と、その結果によって、次に行う処理を指定する命令の二つです。

まず、比較の命令の一つに、CMPというのがあります。

CMP　B

と書けば、Bレジスタの内容とAレジスタの内容を比較しろという命令です。CMPでは、常に比較する数の一方はAレジスタの内容です。

CMP　B

の場合、Aレジスタの内容からBレジスタ

の内容を引き算して、その結果が正か負かの情報を、Fレジスタというところに伝えます。Fレジスタは、1ビットごとに分割された各種のフラグの集まりで、各種のフラグは様々な役割を果たすところです。このプログラムではFレジスタの1ビット分であるSフラグという部分が関わっています。

(A)から(B)を引いた答えが、正か0ならば、Sフラグに0が入り、負ならば、1が入る仕組みになっていて、結局、

(A)≧(B)　ならば、Sフラグ＝0

(A)＜(B)　ならば、Sフラグ＝1

ということになります。(図3—24)

この命令の時は、Aレジスタの内容も、Bレジスタの内容も変化しないで、それぞれ、36、92のままです。

次に、その結果によって、実行する処理を指定する命令の一つに、JPという命令があります。

JP　××××H（××××には、飛んで欲しい命令の入った番地を入れる）

と書けば、Sフラグが0の時、指定された番地の命令に飛び、Sフラグが1の時、すぐ次の命令を実行するようになっています。

図3—23のフローチャートでいえば、③の判断がYESの時、つまり(A)が(B)より大きいか等しい時、Sフラグは0になり、⑤の処理に飛び、NOの時、Sフラグは1になり、すぐ次の④を経てから⑤を行います。

③の判断の部分は、CMPとJPの二つの命令で表わすわけです。

例2のプログラムを図3—25に示します。例1と同じように、プログラム・エリアはメモリの8200H番地から、データ・エリア

```
01  ・ REI－2
02                  ORG     8200H
03                  LXI     H, 8300H    (8200H～8202Hにこの命令が入る)
04                  MOV     A, M        (8203Hに      〃      )
05                  LXI     H, 8301H    (8204H～8206Hに      〃      )
06                  MOV     B, M        (8207Hに      〃      )
07                  CMP     B           (8208Hに      〃      )
08                  JP      820DH       (8209H～820BHに      〃      )
09                  MOV     A, B        (820CHに      〃      )
10                  LXI     H, 8302H    ( 820DH ～820FHに      〃      )
11                  MOV     M, A        (8210Hに      〃      )
12                  HLT                 (8211Hに      〃      )
13                  ORG     8300H
14                  DB      38          (8300Hにこのデータが入る)
15                  DB      92          (8301Hに      〃      )
16                  DB      0           (8302Hに      〃      )
17                  END
```

(行の番号)
ただの目印です

(疑似命令はメモリに入りません)

図３－25　例２のプログラム

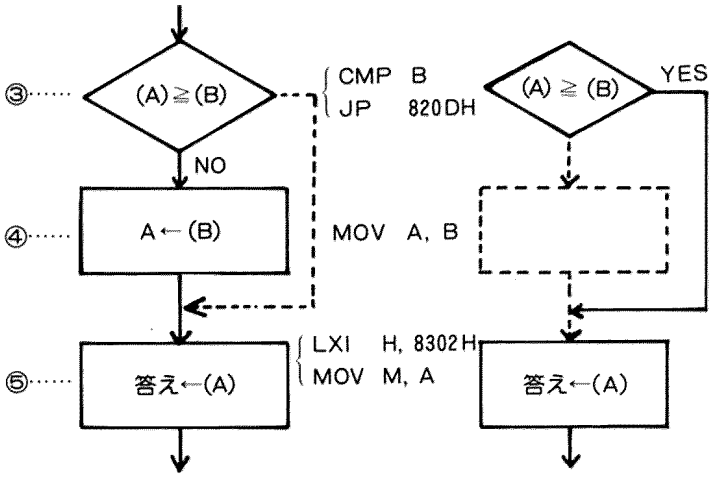


図3－26　例2のフローチャート

はメモリの8300H番地から使っています。

08　JP　820DH

という命令は、07行の比較の結果、(A)が(B)より小さければ、すぐ次の09行の命令を実行し、大きければ820DH番地の命令（ここでは10行の命令）に飛んで、それを実行せよという命令です。

飛ぶと、09行の命令（(B)をAレジスタに移す）を行わないわけですから、11行の命令でのAレジスタの内容は04行の時のままということになるのです。

逆に、09行の命令を行ってからだと、11行でのAレジスタは、06行の命令でBレジスタに入った内容を持っていることになります。

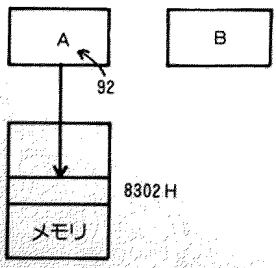
図3—26は、それをフローチャートで表わしています。真中のアセンブリ言語は、すぐ

隣のフローチャートの部分と対応する命令を表わしています。

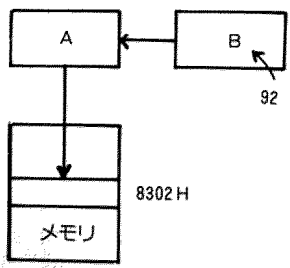
データを、８３００Ｈに36、８３０１Ｈに92と入れた図３―25のプログラムでは、38が92より小さいので、08行の命令を行っても、そのまま順に命令を行っていきます。結局、最初Ｂレジスタに入れた92が、答えの入る８３０２Ｈ番地に入って、38と92を比べた結果、92の方が大きいという答えが出るのです。

逆に、８３００Ｈに92、８３０１Ｈに36を入れたとすると、92はＡレジスタ、38はＢレジスタに入ります。08行の命令を実行すると、８２０ＤＨ番地にある命令、つまり10行の命令に飛んで、09行の命令を実行しないので、Ａレジスタの内容は最初の92のまま、メモリの８３０２Ｈ番地に入れられるのです。

どちらでも、大きい方（92）が、答えを入



（2）（1）の逆だと，命令は10行に飛ぶのでAレジスタの内容は最初のまま，メモリに入る。



（1）Aレジスタに38，Bレジスタに92だと命令は飛ばないので，Bレジスタの内容はAレジスタに入ってから，メモリへ行く。

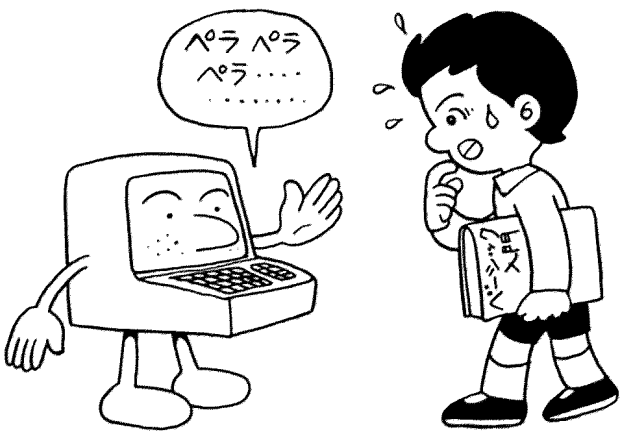
図３－27　答えが出て来る仕組み

れる8302H番地に入るわけで、確かにマイコンは正しい答えを出すわけです。

アセンブリ言語には、ここで示した命令語のほかにも多くの命令語があり、それを組合わせることによって、いろいろな仕事をさせることができます。

マイコンによって違いますが、五〇種から、一五〇種もの命令語があり、これを全部覚えるのは大変です。また、覚えただけではプログラミングはできません。単語を覚えただけでは、英語を喋ることができないのと同じようなことです。

まず、基本的な何種類かの命令語を覚え、それを使って簡単なプログラムをいくつか作ってみることです。それから、徐々に命令語の数を増やして、難しいものに挑戦して行くよりほかに手はありません。



プログラミングは外国語の習得と同じ

外国語の習得と同じことです。

ここでは、ごく簡単な命令語で、わかりやすいように、プログラムを作りました。実際には他の命令語で、もっと短いものを作ることができます。文章にも、冗長な文章と簡潔な文章があるように、プログラムにも冗長なものと簡潔なものとがあります。文章では、どちらがいいとは必ずしもいえませんが、プログラムは簡潔な程いいのです。簡潔なプログラムを作ることは、プログラマの役目の一つなのです。

アセンブリ言語では、プログラミングをするために、ハードウェアの知識が不可欠です。CPUやメモリの構造を知らなければ、アセンブリ言語も使えないわけです。

けれども、ベーシックは、そうではありません。構造を知らなくても、人間の言葉に近い形で、プログラムが作れるのです。

(2) ベーシックによるプログラミング

ベーシックは、高級言語の一つです。普通、インタプリタというプログラムを使って、ベーシックから機械語に解釈し、マイコンに実行させます。

ベーシックだと、プログラミングにかかる労力が非常に少なくなります。例えば、アセンブリ言語で、あれだけ面倒だった足し算の命令は、

```
INPUT A, B
LET C=A+B
```

だけですますことができるのです。

ベーシックの特長は、他の高級言語と同じく、人間の言葉に近い形でプログラミングができ、プログラミングが楽にできること、ほかの高級言語に比べて、文法が単純であるこ

と、などです。逆に難点は、実行速度が遅いことです。

マイコン用に、インタプリタに使うメモリが比較的少なくてすむタイニー・ベーシックというものが開発されていることはまえに述べました。ここでは、最も簡単なタイニー・ベーシックで、プログラムを作ってみましょう。

ベーシックは、インタプリタによって、文法などが異なる場合があります。ここにあげるのは、そのうちの一つで、必ずしもベーシック全体の文法ではありません。

ベーシックを使うには、テレタイプか、テレビブラウン管の付いたタイプライタ型のキーボードが必要です。タイプライタで入力し、プログラムや結果は、印字されるか、ブラウン管に表示されます。もちろん、メモリの中にインタプリタが入っていることも不可欠です。

ベーシックでは、アセンブリ言語で実際のデータのかわりに、(A)だとか(B)だとかでプログラムを組んだように、普通アルファベットの変数でプログラムを組みます。たいてい、A、B、C、を使ったり、データの意味に関係した頭文字を使ったりしますが、アセンブリ言語の時のように、A、B、自体には意味がなく、XでもYでも、一貫してそのアルファベットを使うのなら何でもいいのです。

また、タイニー・ベーシックで扱う数値は、1、2、3、などの整数に限られています。

例3　A、B、C、D、E、の五つの数の平均値を求めるプログラム

フローチャートを書くと、図3―28のようになります。

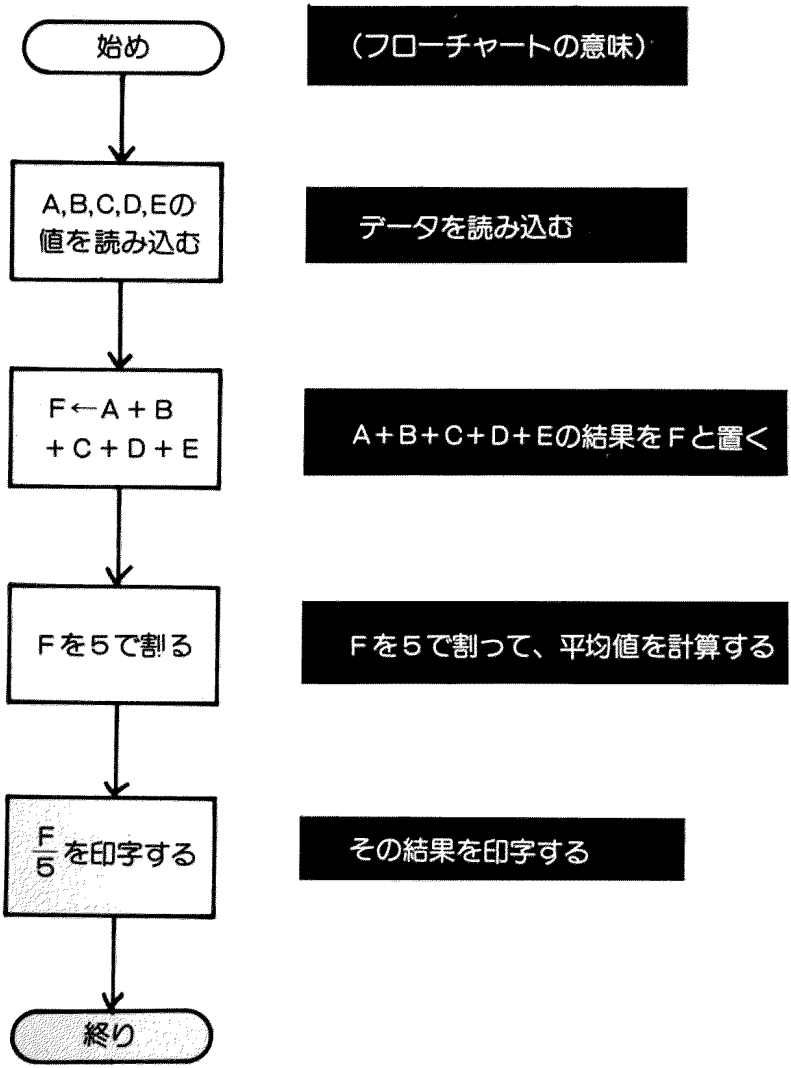


図3－28　例3のフローチャート

```
REI－3

10   INPUT A, B, C, D, E
20   LET F=A+B+C+D+E
30   LET F=F/5
40   PRINT "HEIKINCHI=" ;F

(行番号)
```

図3－29　例3のプログラム

これに基づいて、プログラムを書くと、図3―29のようになります。

行番号は、普通10の倍数で書きます。これは、後で新しい行を挿入しなければならない時に、どこにでも入れられるようにするためです。例えば、図3―29のプログラムに、行番号25の命令を新しく入れると、行番号20と30の間に入る仕組みです。

1　10 INPUT A, B, C, D, E

データを読み込むための命令です。五つの整数をデータとして入れておくと、入れられた順に、A、B、C、D、Eとして読み込んで計算してくれます。

2　20 LET F=A+B+C+D+E

AからEまでを足して、それをFと置けという命令です。ただし、このFが5の倍数になっていないと、出てくる答えは正し

くなりません。タイニー・ベーシックでは、整数しか扱えないからです。もし、ここでのFが5の倍数でなければ、次の行の計算の結果は、小数点以下を切り捨てたものになります。

```
3  30 LET F=F/5
```

前の命令で得たFを5で割って、その答えを再びFと置きます。

アセンブリ言語のところで、AレジスタとBレジスタの内容を足し算すると、その答えがAレジスタの内容になるのと似ています。

ここでは、等号の左のFは、GでもXでもかまいません。もし、Gにしたとすると、次の命令のFのところは、Gに置きかえればいいだけです。割り算は／で表わします。

```
4  40 PRINT "HEIKINCHI=";F
```

"（クォーテーションマーク）で囲んだ部分はこのままの文字が印刷され、；（セミコロン）は間をあけないで次の文字を印字せよという命令で、"で囲まれてないアルファベットは、その文字のもつ値に置きかえられて印字されます。

データが、5、3、7、14、1、の五つだとすれば、この平均値は、

(5+3+7+14+1)÷5=6

ですから、実際に、行番号40の命令で印字されるのは、

HEIKINCHI=6

となります。

タイニー・ベーシックでは、普通、最後のプログラム文を実行したら、停止するようになっていることが多く、停止命令はありません。もちろん、あるものもあります。

このように、ベーシックでは、アセンブリ言語と比べものにならない程、簡単にプログラミングできます。

ただし、できることは、ベーシックのインタプリタによって限りがあり、何でも命令できるわけではありません。それに対して、機械語、アセンブリ言語は、マイコンにできることならあらゆることを命令できます。

次は、もう少し、複雑なプログラムを作ってみましょう。

例4　1＋2＋3＋…＋Nの和を求める。

Nとi	iの値	wの値
7・1	1	0
7・2	2	1
7・3	3	3
7・4	4	6
7・5	5	10
7・6	6	15
7・7	7	21
7・8	8	28

表3－4

Nが例えば5なら、ただ足し算をすれば、いいのですが、Nがどんな整数のデータ（ただし、1より大きい整数）であってもかまわないように、プログラムを作るには工夫が必要です。

フローチャートを書くと、図3―30のようになります。

問題なのは、iとwです。何故、こんな変数を用いるのか説明しましょう。

図3―30のフローチャートを、例えばNを7として、実際にたどってみるとわかるでしょう。iとwの値の変化を表3―4に示します。

iの値が8になった時、③の判断はYESになり、その時のwの値（28）が答えとして印字されます。確かに、

1＋2＋3＋4＋5＋6＋7＝28

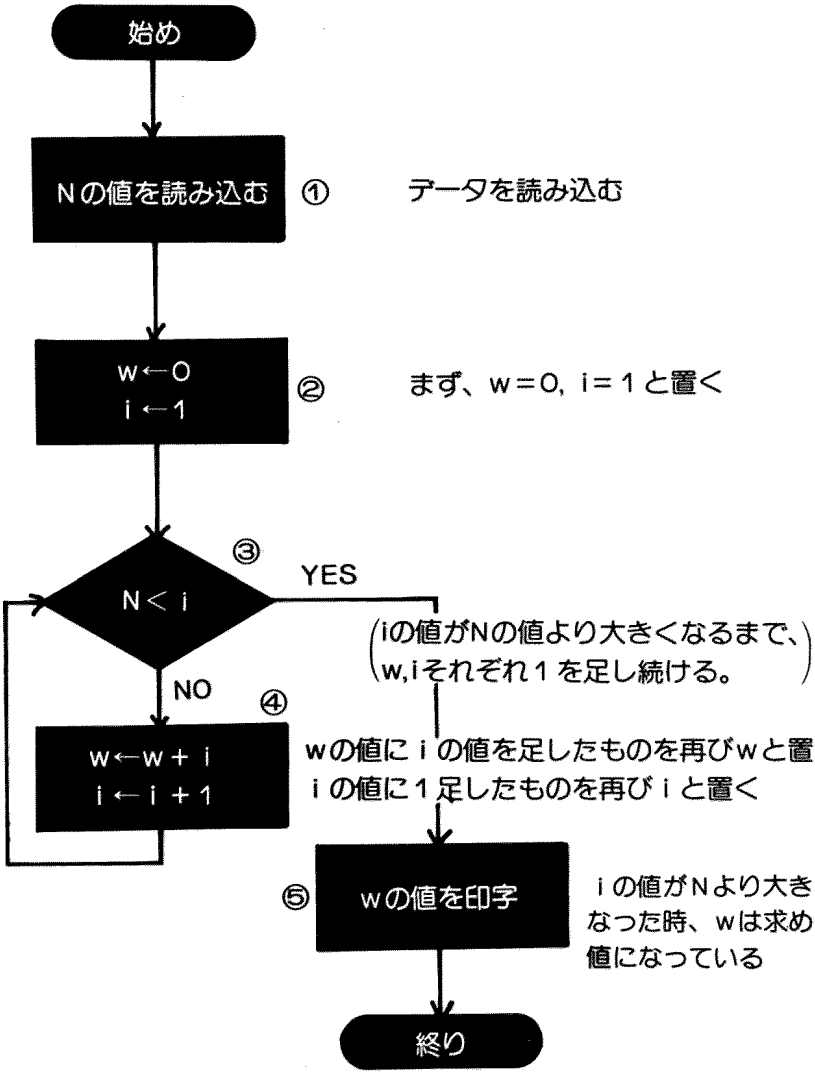


図3－30　例4のフローチャート

なので、正しい答えです。

この操作で、答えが出てくる理由は、④の

w←w＋i　i←i＋1

にあるらしいことはわかるでしょう。ここで、iは、1ずつ増え、wはiの値ずつ増えていく仕組みです。

iが1ずつ増えるということは、表3―4のように、1、2、3、…となって行くことを意味しています。wは最初0ですから、

w←w＋i

の操作を初めて行う時は、iの値（1）がwに入るわけです。

次に、iが2となって、wに足され、次にiが3となってwに足され…、というようにwは、1を足され、2を足され、3を足され、4を足され、5を足され、6を足され、7を足され、次は、iが8になってN（7）より大きいので、wはその状態のまま、⑤の操作を受けます。この時、wは、明らかに、

w＝1＋2＋3＋4＋5＋6＋7

となっていて、結局、1から7までの和、28が⑤の操作によって、印字されるのです。（図3―31）

これは、Nが1でも、3でも、26でも同じことです。

さて、判断と、判断の結果によって行く先を変える命令に、IF…THEN…という文があります。

IF N＜1 THEN 80

と書けば、

N＜1　でない時、すぐ次の行へ、行け
N＜1　の時、行番号80へ、行け

という命令です。

この文と、GOTO文とを組合わせれば、図

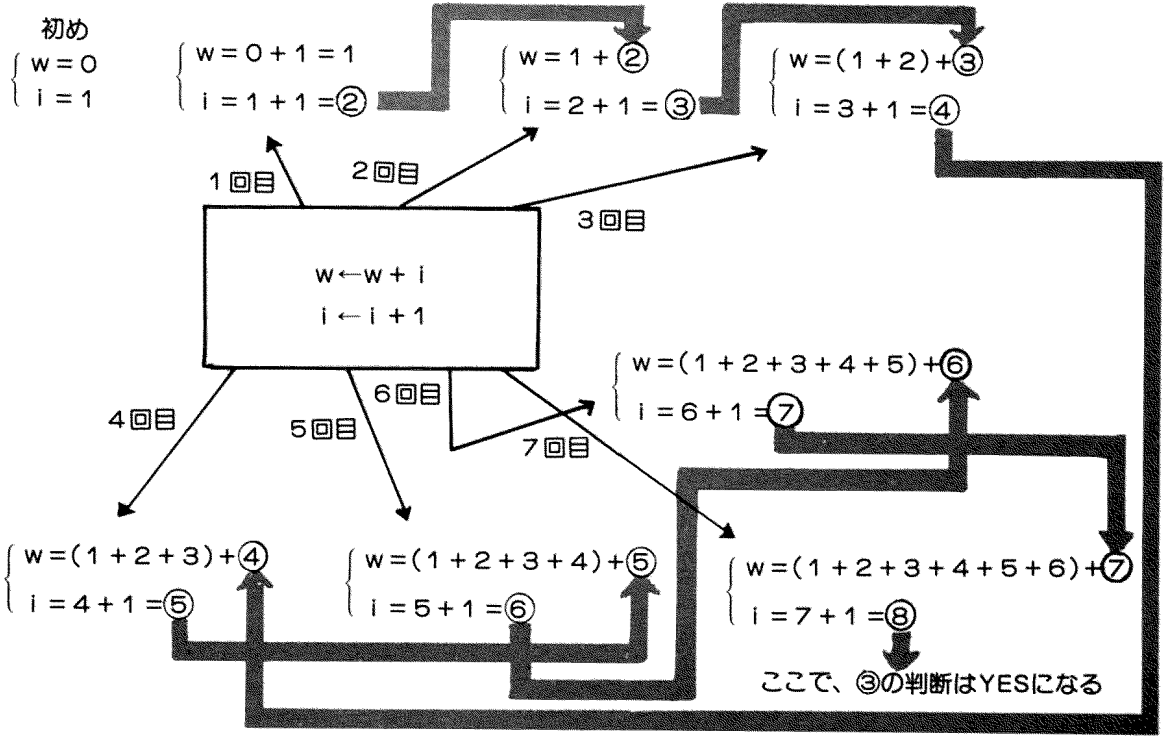


図３－31　Ｎ＝７の時のｉ，ｗの値の変化

```
          REI－4
10  INPUT  N
20  LET  W＝O
30  LET  I＝1
40  IF  N<I  THEN  80
50  LET  W＝W＋I
60  LET  I＝I＋1
70  GOTO  40
80  PRINT  "1  KARA  ";N;"  NO  WA＝";W
```

Nのデータが7の場合、行番号80の命令文によって印字される文字は、

1 KARA 7 NO WA＝28となる。

図3－32 例4のプログラム

3―30のフローチャートの、③、④、の間をくるくる回って、最後に⑤へ行く命令ができます。GOTO文は、

GOTO 40

と書くと、行番号40へ行けという命令になります。

これで、例4のプログラムを書くと、図3―32のようになります。

行番号80のところの意味は、クォーテーションマークだけにはさまれた部分はそのまま、クォーテーションマークにはさまれてない部分は、その文字の値、セミコロンは間をあけないという意味です。

タイニー・ベーシックの最も簡単なものでは、今まで出てきた命令のほかに、GOSUB文と、RETURN文しかありません。

GOSUB 110

と書けば、行番号二〇から始まるサブルーチンへ行けという意味になり、そのサブルーチンの最後の行に

RETURN

と書いて、主プログラムにもどすようになっています。

また、数式の表わし方は、

A×B÷C+D−E

をベーシックでは、

A*B/C+D−E

というように書きます。

最低、これだけのことを知っていれば、例4のような工夫をこらして、いろいろな計算を、させることができるのです。

それ程、ベーシックは簡単です。

ここで述べたのは、もっとも簡単なベーシックですが、基本的にはどんなベーシックでも同じです。後は、数学的な着想の問題です。

デバッグ

できあがったプログラムをマイコンに入れても、答えが出なかったり、間違った答えが出たりすることがあります。

その原因には、二つあります。

① 言葉の使い方の間違い

② フローチャートなど、もともとの考え方の間違い

①の間違いは、正しい言葉に直せばいいだけですが、②の場合は、フローチャートの作成からやりなおさねばなりません。

プログラムの間違いを探すことを、デバッグといいます。デバッグとは「虫とり」の意味なので、デバッグのことを、「虫とり」ということもあります。

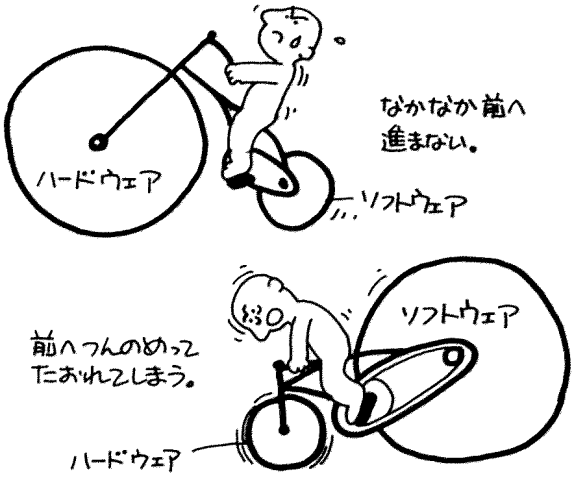


図3－33　ハードウェアとソフトウェア

デバッグは、コンピュータにかけてする場合と、紙に書いたものでする場合がありますが、まず紙でデバッグをするのが普通です。

この章で述べたプログラムや、プログラムを作るための考え方、知識など、をソフトウェアといいます。これは、第二章のハードウェアと同様、コンピュータを扱うためには重要です。

アセンブリ言語や、機械語などは、ハードウェアの知識がなければ使えません。それに対して、ベーシックは、ベーシックの文法さえ知っていれば、プログラミングができます。

けれども、ベーシックを使うにしても、コンピュータの機能、構造の知識がなければ充分ではありません。人間の言葉に近いとはいっても、コンピュータへの言葉なのですから、当然コンピュータの働きと深くかかわってで

きあがっているのです。

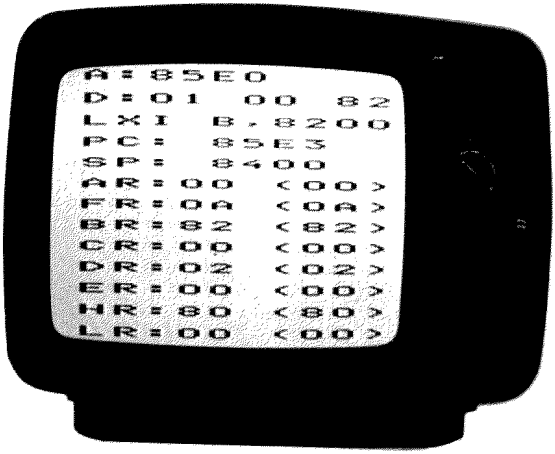
ソフトウェアとハードウェアは、この二つともしっかりしていなければ、コンピュータを深く理解することはできないのです。いわば、自転車の前輪と後輪のようなものです。

ハードウェアの知識ばかりが大きければ、なかなかコンピュータを使いこなすことができませんし、逆にソフトウェアばかり大きければ先走ってしまって、コンピュータは動いてくれません。

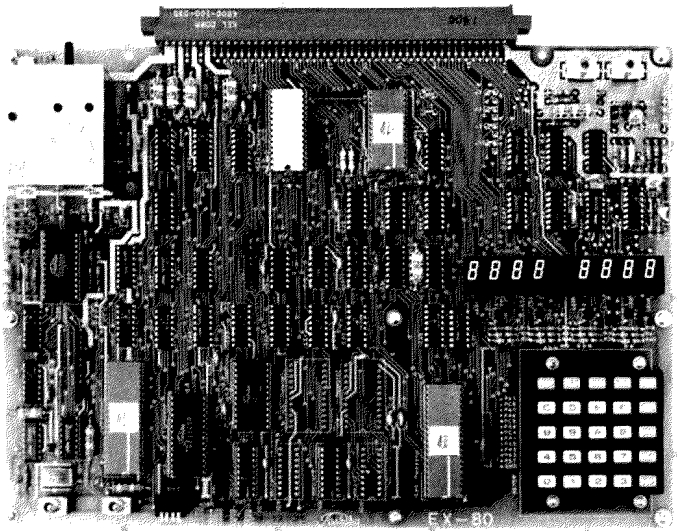
どちらの知識も、それぞれに複雑で入り組んでいます。それを、一歩一歩、自分のものにして行かなければなりません。

少しずつ、難しい本に挑戦して行ったり、マイコン関係の専門誌を読んだり、地道にやって行くよりほかはないでしょう。

単に、遊びとしてマイコンを使いたい人は、人の作ったプログラムなどを使ったりすればいいわけですが、自分で考えて新しい使い方を開発しようという人は、このソフトウェア、ハードウェア二つとも充分に知っていなければできないのです。



★第4章
マイコン・キットの
いろいろ



DO IT YOURSELF

マイコン・キットは、ＤＩＹキットと呼ばれます。これは、

DO IT YOURSELF

（ドゥー・イット・ユアセルフ）

の頭文字を取ったものです。

現在、市販されている国産キットは、アマチュアを対象としたもので、キットの一部をハンダで取りつければ完成するようになっています。ただ、ハンダづけするといっても、簡単ではなく、それなりに細心の注意を必要とすることはいうまでもありません。

したがって、ここでは、ハンダづけなどの組立て作業中に生ずる、二つの代表的なトラ

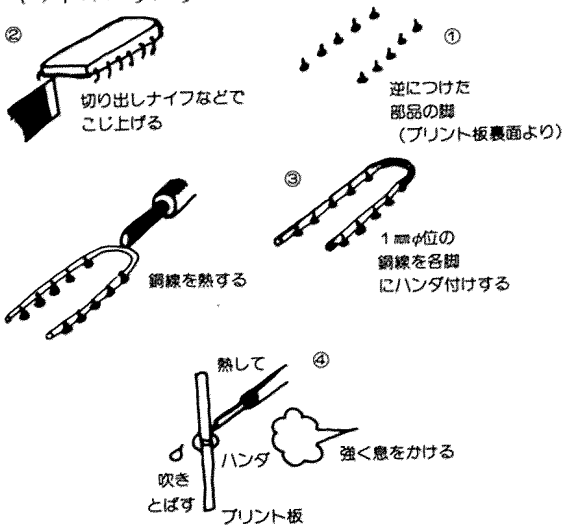


図4－1

ブルの解決策を書いておきましょう。

1　脚の多い部品を逆に取りつけてしまった時の取りはずし方

ICソケット、IC、LED素子など、いわゆるゲジゲジ状の部品の向きは、十分注意して取りつけなければなりませんが、ついうっかりして、逆向きにつけてしまうことが、たまたまあります（左右対称なので、プリント基板の孔に入ってしまうから気づかずにハンダづけしてしまう）。これを取りはずすのは、全脚を同時に熱しないかぎり不可能です。

したがって、この時は、組立て用のハンダゴテとは別に、60ワットぐらいの、やや熱量の大きなコテを用意して、次のような手順で取りはずします（図4—1参照）。

1　まず組立て用のハンダゴテを使って、1ミリ径ぐらいの銅線をプリント基板の裏から

部品の全脚に図のようにハンダづけします。

2　切出しナイフなどの刃をプリント基板と部品の間に当てがい、銅線に大きなハンダゴテを当てて、熱しながら（全脚ともハンダが融ける程度に）刃で部品をこじあげます。（手早く行うこと）

このようにして部品をはずしたら、部品を正しくつけ直すための準備をします。

3　大きなハンダゴテで銅線を熱しプリント基板からとり除きます。

4　部品を外したあとの孔に、組立て用の小ハンダゴテを当てて、ついているハンダを融かし、息を強く吹きかけてハンダを吹きとばします（要は、あらためて部品の脚が入るようにします）。

5　部品の脚の方も組立て用のコテで熱し、ついているハンダを取り除き、曲りを修正し

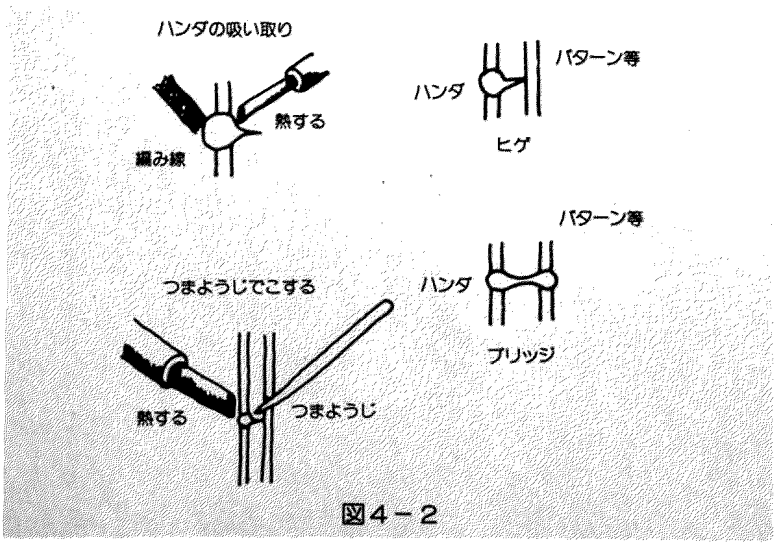


図4－2

ます。

このようにして、あらためて部品を正しい向きにして取りつけ直します。ただし部品の脚が2本とか、ハンダづけが2ヵ所しかしていないうちに逆であることに気がついたのなら、1脚ずつ熱してゆすっていけば取りはずせます。

とにかく、一旦取りつけた部品を外すのは、プロでも苦労しますから、その必要がないよう十分に注意するのにこしたことはありません。

2　ヒゲやブリッジの除き方

ハンダには多小ねばり（温度が低いほどねばる）があり、ハンダづけの際に糸を引いたような形で残り、先が隣りの端子、プリントパターンに触れることがあります。これがヒゲです。そのほかにホコリなどを巻き込んで糸状のヒゲとなることもあります。ハンダゴテにハンダをつけすぎると山をなし、隣りの端子やプリントパターンの方までおよんで橋渡しをしてしまうことがあります。これがブリッジです。ヒゲもブリッジも回路的には短絡になるから除く必要があります。

1　まずハンダの大部分を除きます。簡易的には編み線（シールド線の外側でもよい）をハンダゴテと一緒に当てて、毛細管現象により吸い取らせます。本格的には注射器に似た器具で吸い取るが、それほど大げさに考えなくてもよいでしょう。これでヒゲやブリッジが消滅すればよいが、なおしつこく残ることがあります。

2　しつこく残ったヒゲやブリッジは、ハンダゴテで熱しながら、つまようじでこすり取ります。一見原始的ですが、1ミリ前後の

ヒゲやブリッジ退治には有効な方法です。

●電源を甘くみないこと

購入したマイコン・キットについている説明書に書かれた電流、電圧の数値のみ満たせばよいと考えて、安直に自作の電源ですませうという人が多くいますが、電源の不安定に原因すると思われる誤動作やROMメモリの破壊事故が非常に多くあります。これらを未然に防ぐには安定度に十分配慮した専門メーカーの電源の使用をすすめます。キットメーカーが指定する電源製品を使えば間違いないでしょう。こと電源に関するかぎり費用を惜しんではなりません。

マイコン・キットのいろいろ

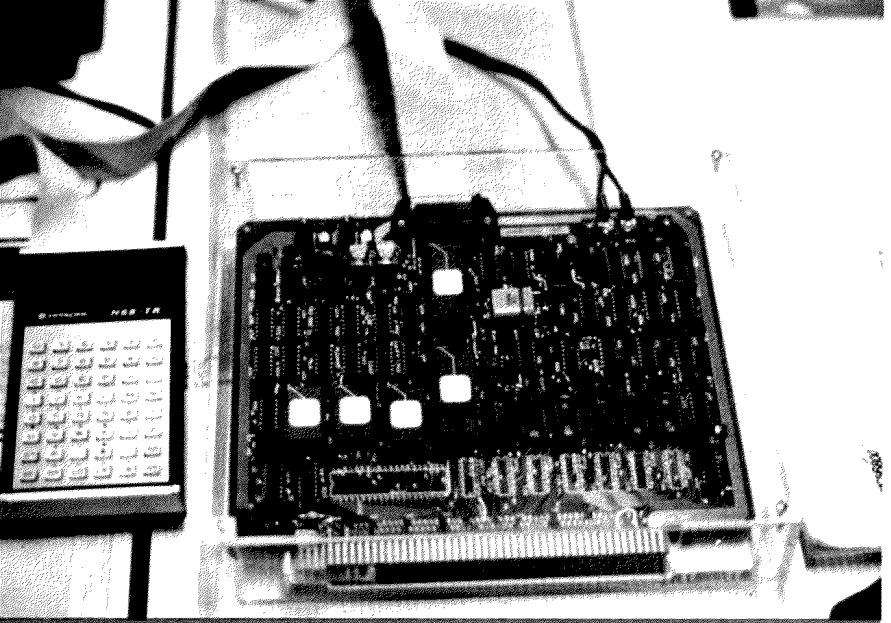
マイコン・キットは、すでに何十種類もの製品がでています。

マイコンは、キットでなくとも、部品をとりそろえて、本当に自分だけのコンピュータを作ることができるのですが、そうするには相当な知識と技術が必要です。

その点、メーカー製のマイコン・キットなら、製作といっても、ほとんどハンダづけだけですし、詳しい説明書（マニュアル）がついていて、ハードウェアもソフトウェアも勉強することができます。メーカーによっては、サービス・センターなどで、買った後でも親切に相談に乗ってくれることもあります。

外国産のキットは、マニュアルが英語で書かれている場合が多いようなので、ここでは国産のおもなキットを紹介することにしましょう。

国産のマイコン・キットは、東芝、日立、



日立　H68/TR

日本電気、パナファコムなどの各社から販売されています。それぞれ、何冊かのマニュアルがついていて、ゲームのプログラムなども掲載されています。

ほとんどが、8ビット型のマイクロ・プロセッサを使用していますが、東芝から12ビット型がでています。パナファコムのLKIT—16は、16ビット型で、マイコン・キットとしては珍らしいものです。

ただ、外観は、むき出しなので、格好よくはありません。けれども、その分、いかにもコンピュータらしい内部構造が見えるので、逆におもしろいかも知れません。

日立　H68／TR

これは、キットではなく、完成品として売られているものです。最小の構成で使うとす

れば全く工作する必要がないのです。作る楽しみを味わいたければ、オプション（あとで、自分の好みで増設できる部分）のスペースを利用すればいいし、製作が面倒な人は、そのまま使えばいいわけです。

写真にあるように、専用のコンソールが付属しています。これを使って、機械語(16進)で入力することも、アセンブリ言語で入力することもできるようになっています。

H68／TRのCPUは、HD46800で、これは、M6800というモトローラ社のマイクロ・プロセッサと同じものです。

8ビットのマイクロ・プロセッサは、大きく分けて三つの系統があります。i8080系と、M6800系と、F8系です。それぞれの系統の名前になっているマイクロ・プロセッサは、外国の代表的なマイクロ・プロセッサなのです。それぞれ、内部構造が異なっているわけですが、そのうちのM6800と、H68／TRのマイクロ・プロセッサHD46800が同じ構造をしているというわけです。

ROMは、4キロバイトのマスクROMで、この中にモニタ・プログラムと、アセンブラが入っています。これのおかげで、そのままでもアセンブリ言語でプログラムを入力することができるのです。

RAMは、1キロバイトが実装されていますが、オプションとして2キロバイト追加することができます。その際、ICソケットがすでに取りつけられているので、RAM用のICをはめこむだけで、メモリを増設できます。

インタフェース部分は、専用コンソール用

と、オーディオ・カセット用のものが実装されています。

オーディオ・カセットは、外部メモリとして使用するもので、同じプログラムをＲＡＭに入れる場合、いちいちキーボードを押して入力しないですむようになっています。また、2台のカセット・コーダを取りつけて、プログラムを編集することもできます。

カセットに納められる情報は、直列処理の情報です。そのために、オーディオ・カセット・コーダのことを、シリアル（直列）Ｉ／Ｏということがあります。コンソールの方は並列、マイコン内部ももちろん並列です。

並列を英語でパラレルというので、情報を並列処理する入出力装置をパラレルＩ／Ｏともいいます。

また、オプションとして、インタフェースＩＣを取りつけられるスペースがあって、そこにＩＣを取りつければ、ほかのＩ／Ｏ装置を取りつけて使用することができます。

最近のマイコン・ショーでの日立の展示物の中に、Ｈ６８／ＴＲにテレビ・インタフェース・モジュール（テレビに画像を映し出すための回路）を取りつけて、テレビゲームを実演しているものがありました。

魚雷ゲーム、ジャンケンゲーム、など普通のゲームセンターにありそうなゲームがたくさん展示してありましたが、こういうゲームも、このマイコンを使ってできます。もちろん、ほかのマイコンも同じことができます。

専用コンソールの表示部（電卓では、数字の映る部分）は、14桁7セグメントけい光表示になっています。

7セグメントとは、七つの切れ端という意

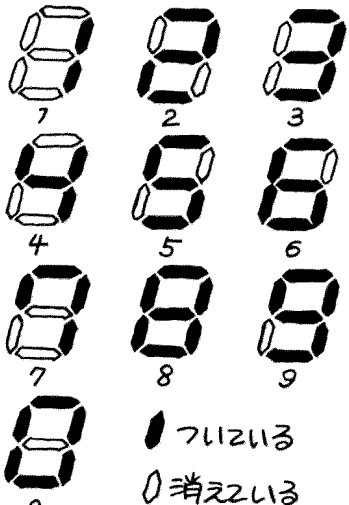


図4－3　7セグメント方式による数字の表示

味で、数字や文字を表示するのによく使われる方法です。電卓は、たいていこの方式を用いているようです。

例えば、数字は、図4―3のように表わされます。

この方式によれば、理論的には一二七種の記号が表わせます。

つまり、この表示方式は七つのけい光ランプが点灯するかしないかで表わされているので、7ビットの情報が表わされるわけですが、$2^7=128$種のオン・オフの組合わせのうち、7つとも消えたものはキーを押さないのと同じことなので、差し引くと一二七種になるわけです。

ただ、数字は、もとの数字とよく似たもので表わせますが、アルファベットや、そのほかの記号などになると違ってきます。例えば、IとかRは、

のように、元の文字とは余り似ていないものになっているので、慣れるのに苦労するかも知れません。

図4―4に、H68／TRの機能ブロック

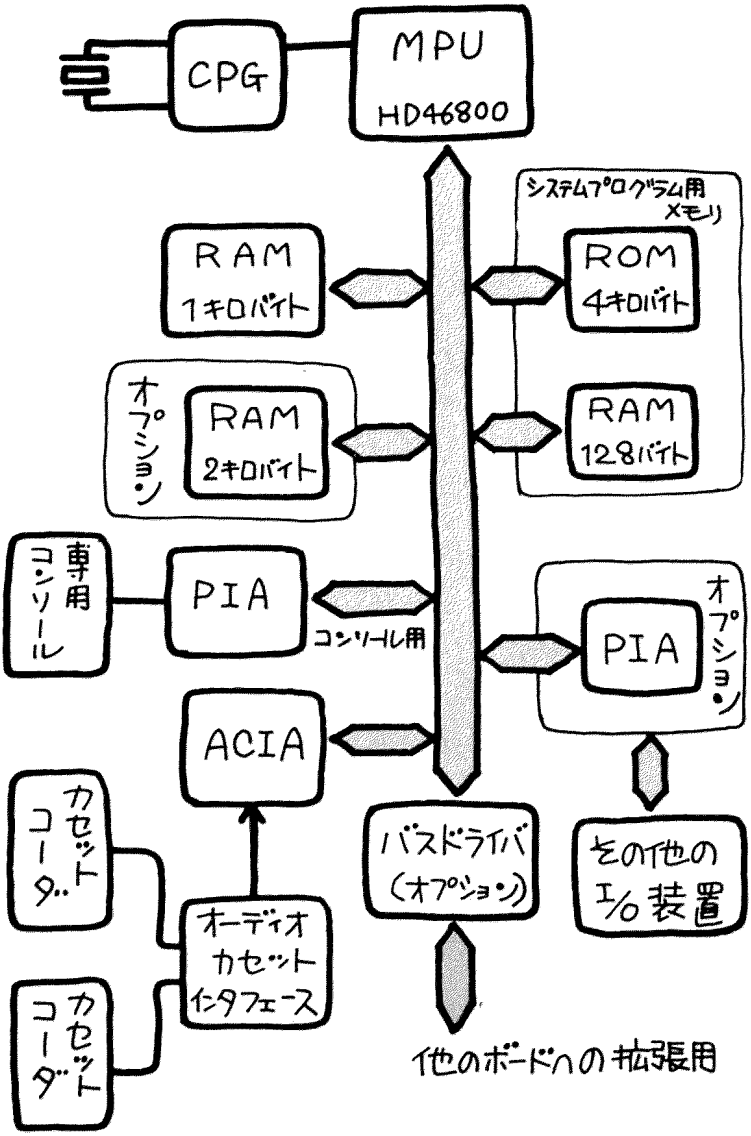


図4－4　H68/TR機能ブロック図

図をあげておきます。

ＭＰＵは、マイクロ・プロセシング・ユニットの略で、マイクロ・プロセッサと同じ意味です。ＣＰＧは、クロック・パルス・ジェネレータの略で、クロック・パルスを作り出す装置です。

ＰＩＡ、ＡＣＩＡは、インタフェースＩＣの名前です。ＰＩＡはパラレルＩ／Ｏとの接続、ＡＣＩＡはシリアルＩ／Ｏとの接続をします。

バスドライバとは、コネクタを使って別の基板とつなぐ時に、コントロールバスやアドレスバス、データバスの信号を強化するためのＩＣです。これらの信号は弱いので、そのままでは多くのＩＣやＬＳＩにつなぐことができないのです。

東芝　ＥＸ－８０

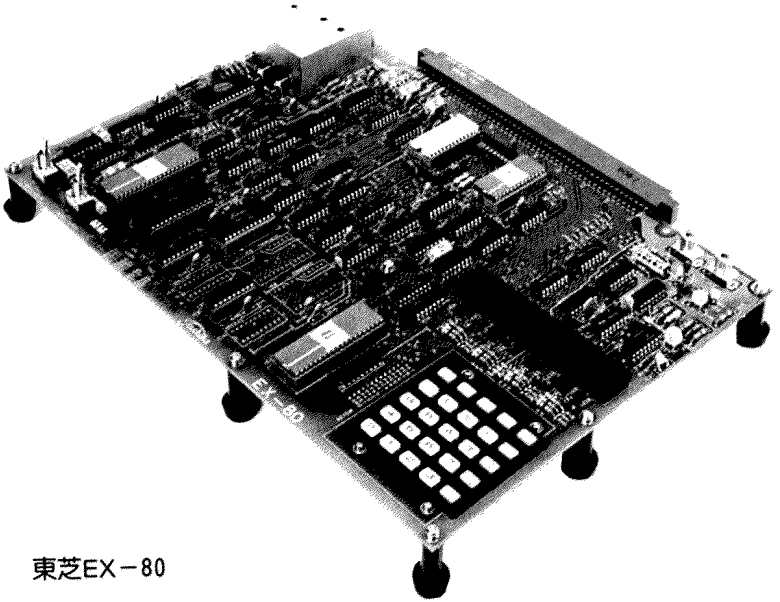
これは、組立用キットです。

ＣＰＵとなるＬＳＩは、ＴＭＰ９０８０ＡＣというマイクロ・プロセッサで、やはり８ビット型です。このマイクロ・プロセッサは８０８０系です。

このキットは、新らしいものの一つで、それだけに、アマチュア向けに設計されているようです。

実装されているＲＯＭは、２キロバイトのマスクＲＯＭで、ここにモニタ・プログラムが書き込まれています。ほかに、２キロバイトまで、ＰＲＯＭを取り付けられるようになっていて、ここに自分の好きなプログラムやデータを入れておくことができます。

ＲＡＭは、１キロバイトのものが実装され



東芝EX－80

ています。オプションとして、さらに1キロバイトのRAMが増設可能です。

インタフェースは、オーディオ・カセット用のインタフェース回路で、カセットを外部メモリとして使用できるほか、テレビ用のインタフェース回路も実装されています。これを使って、家庭用のテレビに、図形や文字を映すことができるのです。

キーボードは、基板に直接取りつけられていて、16進の機械語を入力できるようになっています。アセンブリ言語は、このキーボードだと使えません。アセンブリ言語を使うには、外部に、別のキーボードを取りつけなければなりません。

しかし、16進キーボードで入力したプログラムを、テレビ画像にアセンブリ言語の形で映し出すことができるので、プログラムの確

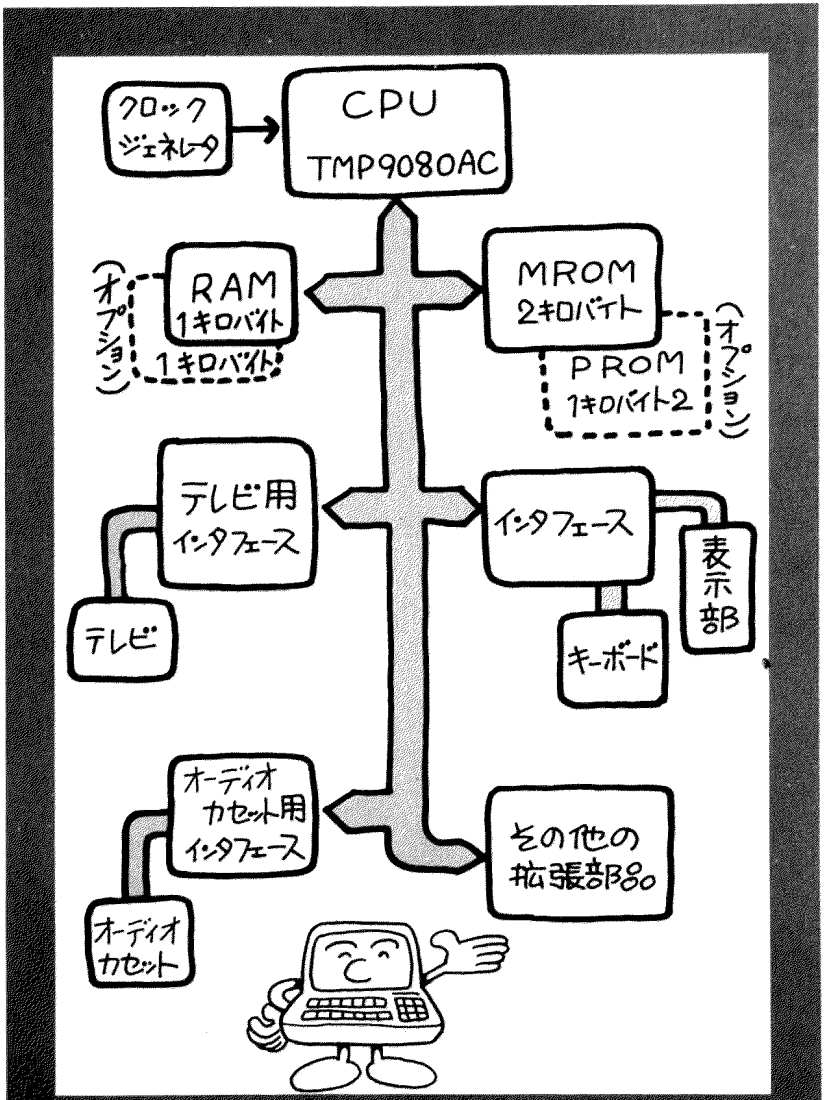


図4－5　東芝EX－80ブロック図

認が比較的容易にできます。

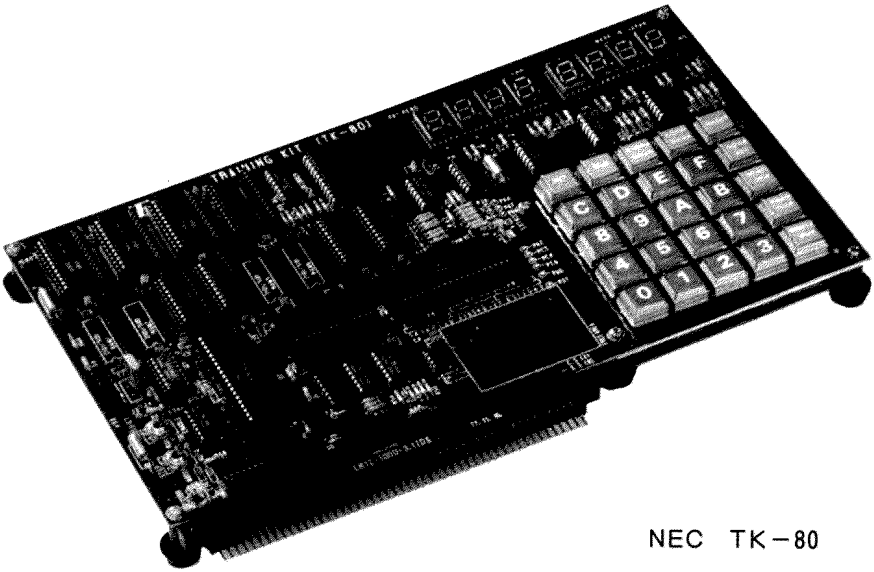
表示部分は、16進数八桁分を、7セグメント方式で表示するようになっています。上位四桁分がアドレス、下位四桁分がデータ用（プログラム情報とデータ情報）になっています。

EX—80の機能ブロック図を図4—5にあげておきます。

ほかに、東芝からは、12ビット型のマイコン・キットが出ています。EX—0、EX—5、EX—10と呼ばれる三つです。

これらは、いずれもキーボードがついていないので、12個の小さなスイッチを一つ一つオンにしたり、オフにしたりしてプログラムを入れなければなりません。別売の16進キーボードを接続して、16進数の機械語で入力することもできます。

しかし、この12ビット型のキットは、いずれも使用者の自由にまかされている部分が多く、あまり初心者向きとはいえません。ただ、うまく使いこなせば、12ビット型なので、8ビット型より複雑な仕事をさせるのに適しているといえます。



NEC　TK－80

日本電気（NEC）TK－80

使用しているマイクロ・プロセッサは、μPD8080ADというもので、8080系です。TK－80より少し性能と価格の低いTK－80E（エコノミータイプのE）では、μPD8080AFCを使っています。

TK－80と、TK－80Eとでは、CPUとメモリに少し違いがあるだけで、あとは価格以外全く同じものです。

ROMは、七六八バイト実装されていて、ここにモニタ・プログラムが入っています。ほかに、二五六バイトのROMをさらに組込むことができます。

RAMは、五一二バイト実装されていて、さらに最大1キロバイトまで拡大することができます。

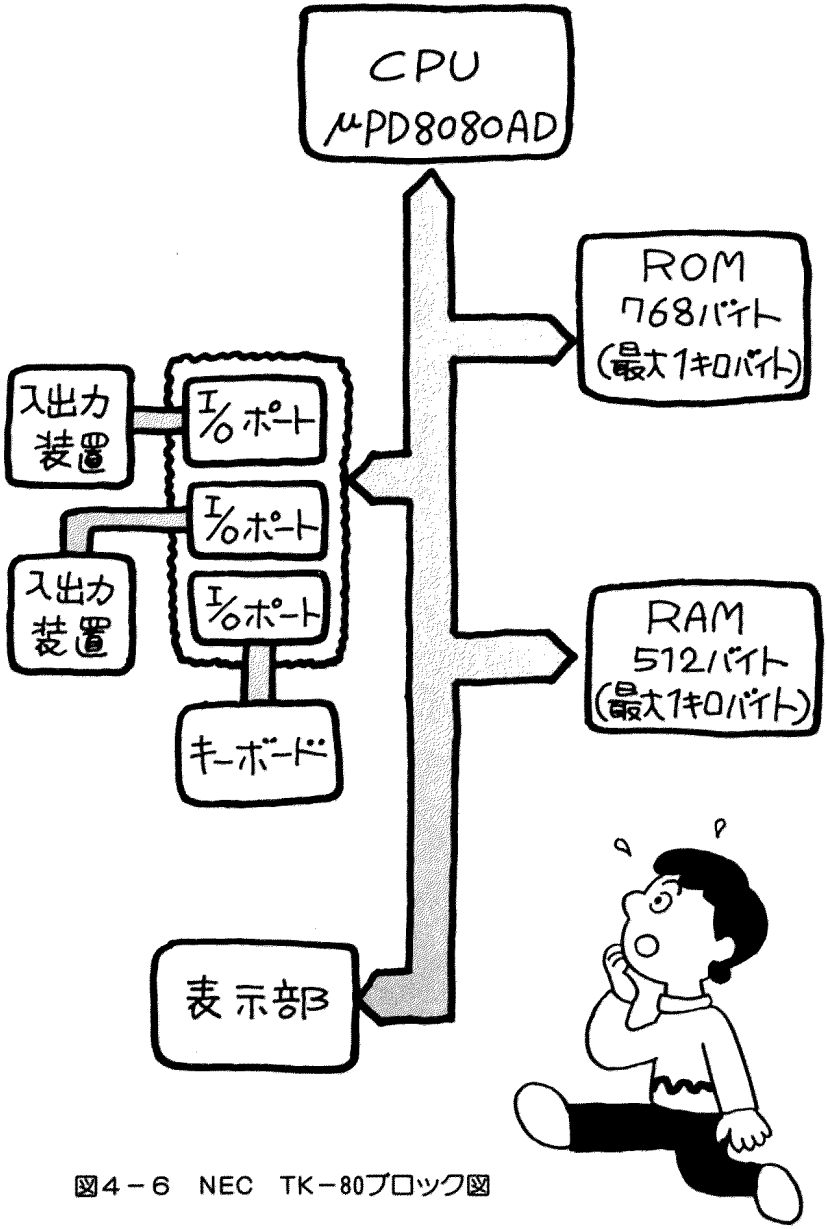


図4－6　NEC　TK－80ブロック図

キーボードは、16進入力用で、東芝のEX－80の場合と同じです。

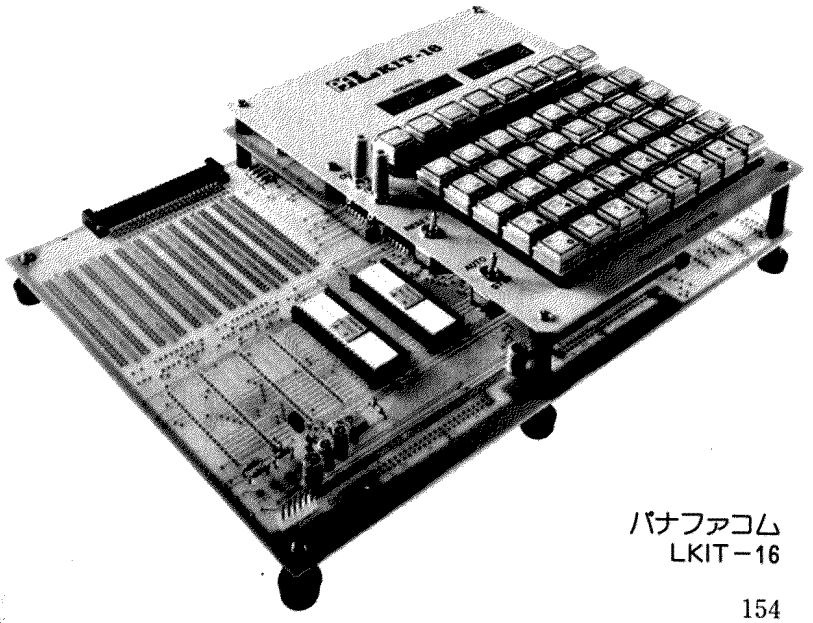
様々なインタフェース回路を取りつけて、テレビやオーディオ・カセットコードと接続することができる点は、まえに書いたものとあまり変りありません。

ほかに、TK－80BS（ベーシック・ステーション）というボードを接続させれば、ベーシックが使えます。タイプライタ型のキーボードがついているので、いろいろな文字、図形を、テレビ画面に映し出して、テレビゲームや、小数計算などができます。

パナファコム　LKIT－16

日本では唯一の16ビット型マイコン・キットです。

CPUは、MN1610という16ビット並



パナファコム
LKIT－16

列処理のマイクロ・プロセッサです。

ROMは、1キロワード（16ビット型なので、1ワード＝16ビット＝2バイト）が実装されていて、モニタ・プログラムが入っています。さらに、ROMを1キロワード増設することができます。

RAMは、五一二ワード分が実装されていて、さらに五一二ワードを増設することができます。

キーボードは、アセンブリ言語用の48個のアセンブラキーでできていて、そのままアセンブリ言語を使用できるようになっています。

表示部は、ほかのものと同じ仕組みです。

また、タイニー・ベーシックのインタプリタをメモリに入れれば、この本体のキーボードでタイニー・ベーシックを使うことができるようになっています。

16ビット型なので、命令語が8ビット型のものより簡単になっていて、高度なプログラムに適しています。

拡張用のメモリボードや、いろいろなインタフェース回路を取り付ければ、家庭用のカラーテレビに色付の図形や絵を映し出したり、テレビゲームで遊んだりできます。

その点では、ほかのマイコンでも同じです。

マイコン・キットを拡張してどのように応用するかは、マイコンを扱う人のアイデア次第であることには変りありません。

そのほか

今まであげたもののほかにも、たくさんのマイコン・キットが販売されています。キットになっていない完成品のものも含めると、数十種類以上になるでしょう。ただし、そう

なると、ミニコンとほとんど変らない高級で高価なマイコンも含まれます。また、簡単なものなら、3万円以下のマイコン・キットもあるのです。

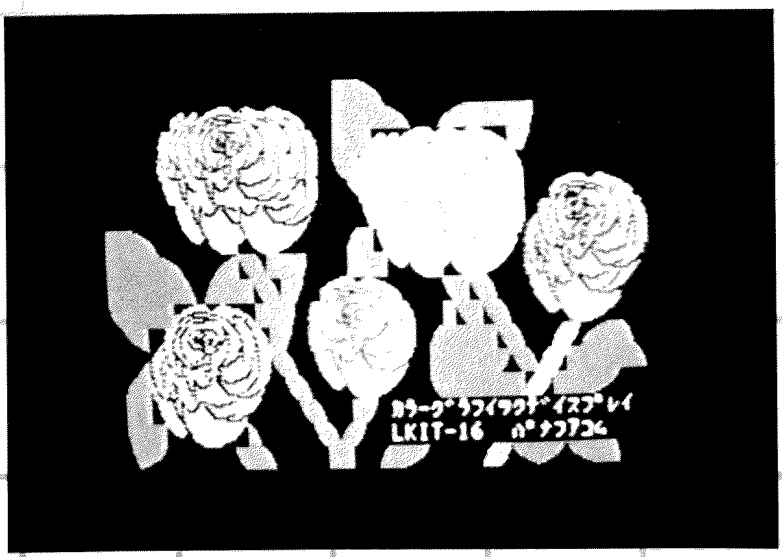
シャープのMZ—40K「マイコン博士」は、4ビット型でスピーカーを内蔵しているので自動演奏などで遊ぶことができるようになっています。子供向けなので、説明書も親切にできています。

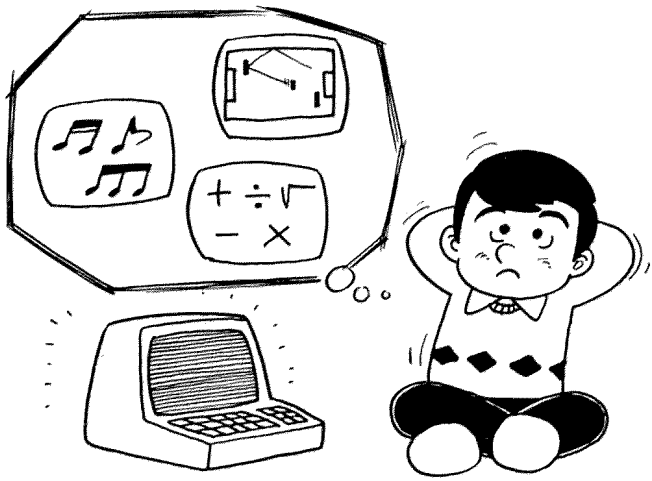
また、最近発売されたCOMKIT8060は、そのままでテレビに接続してベーシックが使える便利なものです。

ほかの8ビット型マイコン・キットでは、富士通のLKIT—8、ロジックシステムズのMP—80、インテル社のSDK—85などがあります。

マイコン・キットとして売られているものは、大体そのままではあまりおもしろいことはできません。コンピュータを手元において、実際に動かしてみるという楽しさは味わえるでしょうが、テレビ・ゲームやそのほかの応用に使う場合は、インタフェースを取りつけたり、メモリを拡張したり、しなければならないのが普通です。つまり、マイコン・キットとしてのものは、基本的なものだけなのです。

★第5章
マイコンに何が
できるか





マイコンに何ができる？

人工知能

人間は、よく考えると大変複雑なことをしているということはまえに書きました。

あいまいな言葉を理解したり、基本的なルールを教えてもらうだけで、複雑なゲームで遊ぶことができたり、まったく見たこともない数学の問題の答えを出したり、今までになかったものを発明したりします。それは人間に知能があるからです。知能とは、判断し、推論し、新しいものを思いつく能力です。

機械には普通知能はありません。知能の仕組みはこみいっているので、それを機械に与えるのはほとんど不可能なのです。

ところが、コンピュータが発達するにつれて、機械（コンピュータ）も人間の知能に似

た働きができるようになってきました。

例えば、コンピュータにチェスをさせることができるのです。チェスは、日本の将棋に似た西洋のゲームですが、将棋でも、やってみればわかるようになかなか難しく、戦局を判断し、自分に有利なように駒を動かすのは容易ではありません。

それを機械であるコンピュータがするのですから、コンピュータも知能を持っているのだといえます。

コンピュータのことを「人工知能」ということがあるのは、そういう意味なのです。

テレビマンガで、人工頭脳を持った人間のようなロボットが活躍したりしていますが、あのようなロボットも決して不可能だとはいいきれなくなっています。

まだ、コンピュータの歴史は始まったばかりです。コンピュータの可能性は無限にあって、まだまだ何ができるのか、よくわかってない部分がたくさんあります。

人間も、生まれたばかりの時は、何にもわからない、ただの物体のようなものです。

頭の中は、脳細胞がばらばらに入っているのですが、年をとるにつれて、それぞれの細胞が互いに連結しあい、成人するころには完全な形で人間としての知能を発揮できるようになっているのです。その過程では、言葉を少しずつ覚えて行き、だんだん新しいことを知って行くにしたがって、脳細胞が複雑に結びつきあい、多くの記憶をとどめて行くのです。

それは、コンピュータでいえば、ちょうど多くのプログラム、多くのデータを入れて行く過程に似ています。ただ、コンピュータは

人間程の複雑さをまだ持っていないので、今のところ、人間のようにはなれないのです。

人間が、子供の頃は一体どんな人間になるのかわからないように、コンピュータも一体何ができるのかわからないのです。

コンピュータに何をさせるか、それは人間の側の問題です。コンピュータより高い知能を持った人間が、いわばコンピュータを教育しなければなりません。

組立てたマイコン・キットは、そのままでは、ごく簡単な人工頭脳にすぎません。メモリも小さいし、目や耳や手足にあたる入出力装置もほとんどありません。

メモリを増やし、いろいろな入出力装置を取りつけても、そのままでは何にもしてくれません。させたい仕事をプログラムに組んで、必要なデータと一緒に入れてやらなければならないのです。

工夫次第では、マイコンの可能性は無限です。マイコンに何ができるかは、使う人間の工夫の問題です。

その人間の工夫の結果、初めてマイコンは人工知能となるのです。

マイコンの応用

マイコンの応用分野は無限にあります。

何しろプログラム次第でいろいろなことができるのですから、今まで人がしていた仕事の一部をマイコンに肩がわりさせて機械に組込むなどすれば、どんなものにでも応用できるのです。

最近は、マイコンを組入れた電気製品が出てきています。

産業での応用

肉屋などで使っている秤にもマイコンの入っているものがあります。いわゆる電子秤というもので、皿に肉をのせると、その重さと値段をデジタル表示してくれるものです。

カメラにもマイコンが組込まれたものがあります。自動露出装置つきのカメラは今までにもありましたが、それをマイコンがもっときめ細かくやってくれるものです。

ミシンにもマイコンが入ったものがあります。テレビにまでマイコンが入りこんで、自動的にチャンネルを切りかえてくれるものがあります。

オートメーション工場でも、銀行の自動預金支払機にも、自動販売機にも、パチンコ屋にも、スーパーマーケットのレジにも、いた

るところにマイコンが入りつつあります。

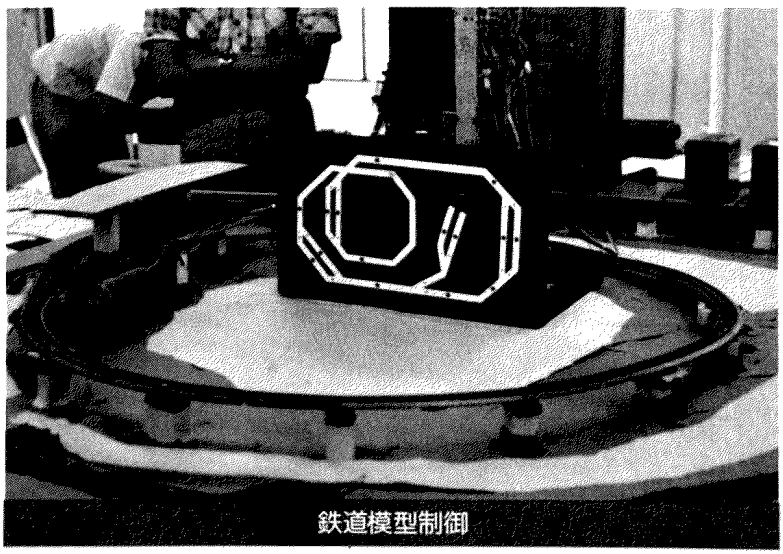
自動販売機などは、マイコンでなくとも簡単な仕組みで動かすことができるのですが、今までの自動販売機だと、二〇〇円入れて、80円のものをまとめて二つ買うなどということはできなかったのに、マイコンが入っている自動販売機では、まるで店員さん相手に買物をしているようなことができるのです。

マイコンで遊ぶ

だいたい、趣味としてのマイコンは遊ぶのが主な目的です。

高級な数学の問題をマイコンに解かせることも一種の遊びですが、数学で遊ぶのは、数学的知識を持っていないと面白くないかも知れません。

自動演奏



鉄道模型制御

スピーカーのついたマイコンで、簡単に電子オルゴール遊びができます。

演奏させたい曲のメロディのデータをプログラムと一緒に入れてやれば、電子音で演奏してくれます。最近、はやっている風船割りゲームのメロディなどを入れてみるのも面白いでしょう。

普通、単音メロディしかできませんが、二音で重奏することもできるようです。

また、シンセサイザーなどに接続して自動演奏をさせることもできます。

自動演奏でなくとも、コンソールのキーボードをオルガンの鍵盤のように使って、メロディを演奏することもできます。

鉄道模型の制御

新幹線がコンピュータで制御されているように、鉄道模型を自動的に駅に止まらせたり、ポイントを切りかえたり、貨車の入れ替えをしたり、実際の鉄道の制御をまねた遊びができます。

向かいあって走る二つの列車が、ふたまたに分かれたポイントで出会った時は、先に来た方が一方の線路で止まったまま、もう一方の列車を待っていて、それが通り過ぎたのを確認してから、再び走り出したりするのです。

デジタル時計

デジタル表示装置を使って時計を作ることができます。

ただ時間を刻むだけでなく、世界各地の時間も表示するようにしたり、ストップウォッチとして使うようにしたり、様々な応用が考えられます。

電気がきている限り、遅れたり進んだりしないので、時計としても優秀なものです。

テレビゲーム

テレビをディスプレイ装置（表示装置）に使った、様々なゲームが考えられます。マイコンによるゲームの中でも、テレビゲームが主流になっています。

ゲームセンターにおいてあるようなテレビゲーム以外にも、自分で独自のゲームを考案できるので、遊ぶ楽しみだけでなく、ゲームを考え出す楽しみもあります。

テレビゲームとして遊べるゲームを二、三紹介しましょう。

(1) ライフゲーム

ライフゲームとは、J・H・コンウェーという人の考案による、生物社会を象徴したゲームです。

生物社会では、多くの生物が生存競争をしています。そこでは、生き残るものも、死んでしまうものも、新しく誕生するものもあります。この生物社会の状況を、マス目に書き入れたいくつかの丸で表わしたものが、ライフゲームです。ルールは三つしかありません。マス目の中の丸が一つの生物という設定ですが、一つのマス目の周囲にほかの丸（ほかの生物）がいくつあるかで、生存、死滅、誕生が決まるのです。

① 生存の条件

一つの丸の周囲に、ほかの丸が二つか三つあれば、その丸は生き残る。

② 死滅の条件

一つの丸の周囲に、ほかの丸が一つ以下しかないか、四つ以上あれば、その丸は消滅する。

③ 誕生の条件

丸の入ってないマス目の周囲に、三つ丸

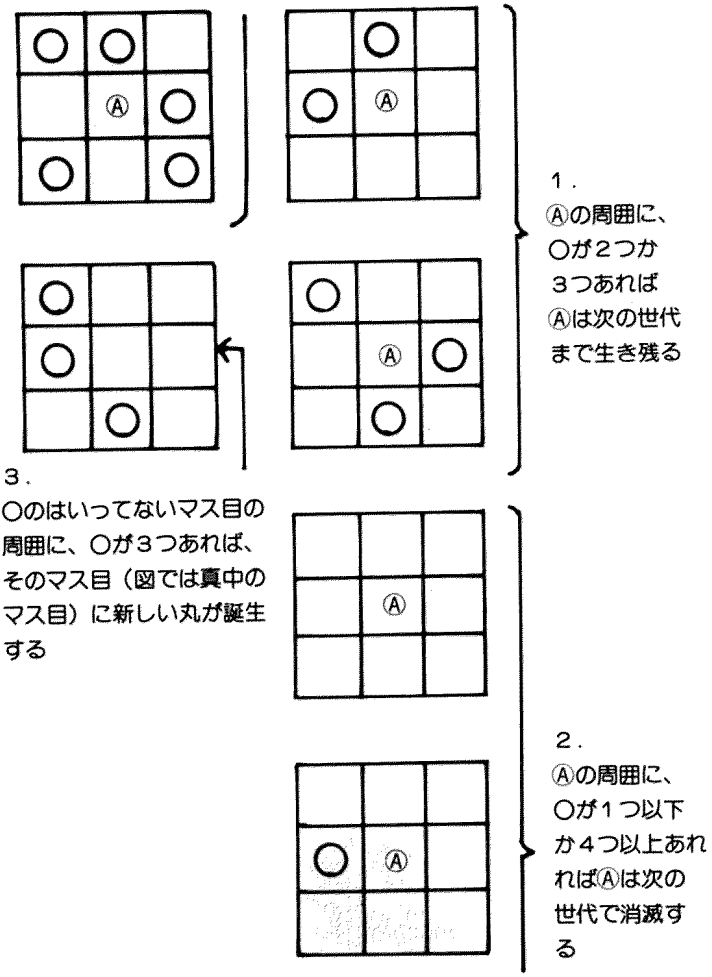
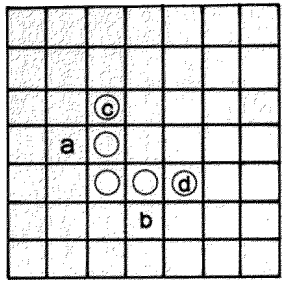
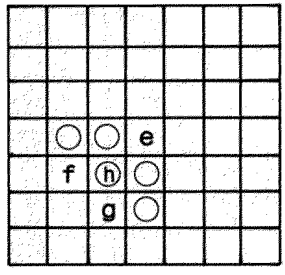


図５－１　ライフゲームのルール



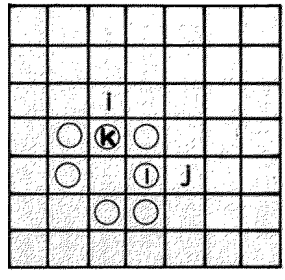
第一世代

aとbのマスに新しい丸が誕生します。cとdの丸は死んでしまいます。



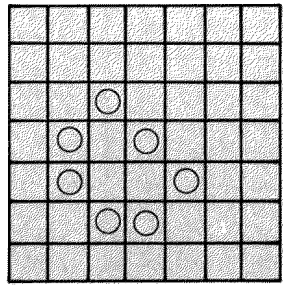
第二世代

e、f、gのマスに新しい丸が誕生し、hの丸が過密のために死んでしまいます。



第三世代

iとjのマスに新しい丸が誕生し、kとlの丸は死んでしまいます。

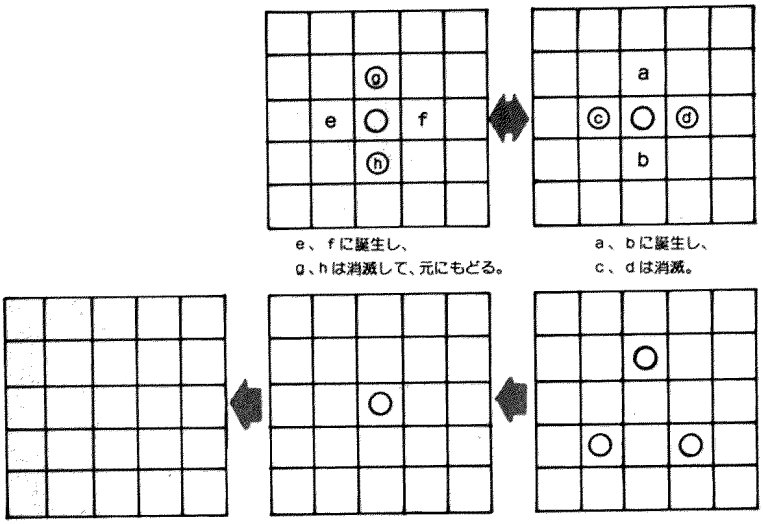


第四世代

すべての丸が生存の条件を満たしています。また、どのマスにも新しい丸が誕生することはありません。

このままの状態が永遠に続くことになります。この状態を定常状態といいます。

図5－2　ライフゲームの簡単な例



（下）図5－4　全て消滅してしまうパターンの例　（上）図5－3　同じパターンを繰り返す例

があれば、そのマス目に新しく丸が誕生する。

この三つのルールにしたがって、次々に操作を繰り返すと、丸の集まりは色々な形に変化して行きます。中には定型的なパターンを示すものや、どんどん増殖して行くものや、全て死に絶えてしまうものなどがあります。ただし、この三つのルールは、ある世代での丸の集まり全てに同時に適用し、それによって新しい形になった集まりに再び三つのルールを同時に適用するようにします。

これは、根気さえあれば、方眼紙に次々に書いていって遊んでも面白いものです。絶好の暇つぶしになります。しかし、テレビに映し出して、てっとり早く世代を進めて行く方が変化がよくわかって面白いでしょう。

最初の世代で、どのように丸をおくかで、

その生物社会の運命が決まってしまうのですが、いくつかの有名なパターンが発見されています。

図5―2で簡単なものを実際に示します。最初のパターンをいろいろ組合わせて試してみると、たいていのパターンは、図5―2のように安定してしまうか、あるいは消滅してしまうか、図5―3のように同じことを繰り返すかにいたります。それらをまとめて、定常状態と呼びます。

中には、いくらやっても無限に丸を増やし続ける「グライダー銃」と呼ばれるものや、いろいろなパターンを吸収して、自分はもとにもどってしまう「食いしん坊」と呼ばれるパターンなど、おもしろい名前のついたものが発見されています。

ライフゲームについては、中公新書「数理パズル」一〇八ページから一一三ページにもう少し詳しく書かれています。

(2) スタートレック・ゲーム

スタートレックとは、アメリカのテレビドラマの名前で、それをゲーム化したものが、これです。アメリカでは大流行しているそうです。

プレイヤーは、宇宙船エンタープライズを操作して、敵であるクリンゴンを撃滅するのがゲームの目的です。一定の期限までに、クリンゴンを全滅させれば勝ち、期限がきてもクリンゴンがまだ残っていれば負けです。

宇宙船エンタープライズは、探知器を使って宇宙の様子を探りながら、光子魚雷と光子銃を使ってクリンゴンを攻撃します。

燃料には限りがあって、時々補給のために宇宙基地に立ち寄らねばなりません。クリンゴンの方も、宇宙船エンタープライズを攻撃

してくるので、もしクリンゴンにやられた場合、修理のために手間取ることになります。

このような条件のもとで、宇宙戦争がテレビブラウン管上で繰り広げられるのです。

そのほか、マージャン、トランプ、サイコロ、競馬など、いろいろなゲームが考えられます。

テレビでは、実際のものの形や動きをマネして映すことができるので、図形の動きで表現できるゲームなら、ほとんど何でも可能です。

このように、実際の物の形や動き、実際のゲームなどを、コンピュータでマネすることを、シミュレーションといいます。シミュレーションとは、マネするという意味の英語です。

(3) その他

モールス・コンバータ　モールス信号を、文字に直すモールス・コンバータとして、マイコンを使うことができます。解読されたも

A	・—	N	—・
B	—・・・	O	— — —
C	—・—・	P	・— —・
D	—・・	Q	— —・—
E	・	R	・—・
F	・・—・	S	・・・
G	— —・	T	—
H	・・・・	U	・・—
I	・・	V	・・・—
J	・— — —	W	・— —
K	—・—	X	—・・—
L	・—・・	Y	—・— —
M	— —	Z	— —・・

図5－5　モールス信号とアルファベットの対応

ールス信号は、テレビブラウン管や、印字機などで文字として出力されます。

モールス信号は、信号と文字の対応さえ覚えておけば簡単なように思えますが、実際はなかなか難しいものです。わずかの間が、重大な違いをもたらすのです。

電子ロック　鍵の開閉をマイコンで制御するものです。例えば、暗号を決めておき、正しい暗号を入力すればロックは開き、何回か違った暗号を入力すると非常ベルが鳴るようにしておくと、防犯にもなります。

カレンダー　印字機を使って何年分かのカレンダーを印刷させることもできます。もちろん計算はマイコンが行います。

その他、アイデアと道具と材料さえあればマイコンの応用は、無限に開けています。

一応、マイコンに関する説明はこれでおわりです。あとは、それぞれのマイコン・キットに付属するマニュアルを読んで、マイコンへの理解を深め、知識を増やして下さい。

最後に、第六章として、いくつかの重要なマイコン用語について、説明してあります。マイコン・キットのカタログや、マニュアルを読んだり、新しい知識を理解する上での参考にして下さい。

第5章までの部分で説明していないところなど、補なってありますので、辞書のように引くのではなく、頭から読んで行くことをすすめます。

少し、難しいかも知れませんが、この程度の用語は必要です。少しずつ理解しながら、読み進めて下さい。

★第6章
マイコン用語の解説

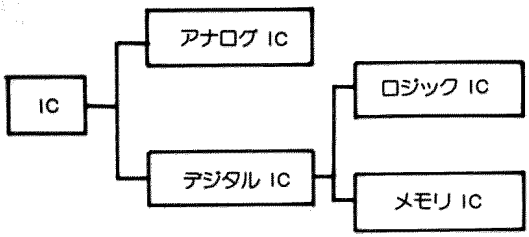
―ア―

アクセス・タイム＝ＣＰＵがメモリに書き込み、読み出しを要求する信号を出しても、メモリはすぐに応答できるわけではなく、一定の時間がたってから実際の書き込み・読み出しができるようになります。この遅れ時間をアクセス・タイムと呼びます。

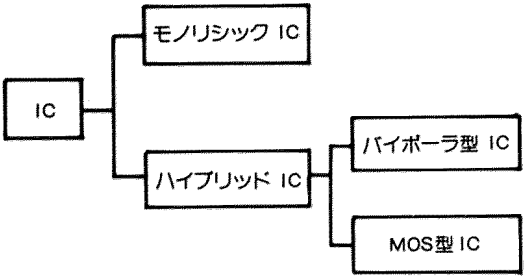
アセンブラ＝機械語は、普通の人には理解できないので、わかりやすく覚えやすいように、文字や記号を使って表わすと便利になります。プログラムのこのような書き方をアセンブリ言語と呼び、アセンブリ言語で書かれたプログラムを機械語に直すソフトウエアのことをアセンブラと呼びます。（普通はアセンブリ言語のこともアセンブラと呼びますが、この言い方は、本当は正しくありません）そして、アセンブリ言語から機械語に直す作業をアセンブルと言いますが、アセンブルを実行するコンピュータとアセンブルされた（機械語）のプログラムを実行するコンピュータが同じ場合にはセルフ・アセンブラと呼び、違う場合にはクロス・アセンブラと呼びます。「機械語」の項参照。

アドレス＝日本語で「番地」。コンピュータがデータを記憶しておくための場所は一つだけではないので、それらを区別するために番号を付けておかなければなりません。アドレスとはこの番号のことです。実際にプログラムの中で、ある番地を指定するにはいろいろなやり方があって、このやり方の

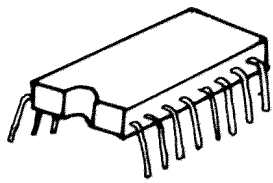
ICの役割による分類



ICの作り方による分類



マイコンで使うICはだいたい
1．役割による分類では
デジタルIC
2．作り方による分類では
モノリシックICで
下図のような形をしています。
（DIP型）



ことをアドレシング・モードと呼びます。

アナログとデジタル――例を使ってごく簡単に説明すると、時間を表わすのに、針の角度という連続的な量を使えばアナログ時計であり、数字というとびとびの値を使えばデジタル時計となります。

I／O――「アイ・オー」と読みます。「イチ・ゼロ」ではありません。インプット／アウトプットの略で、コンピュータの入出力のこと。

IC――小さなパッケージの中にトランジスタやダイオードなどを数十個から数千個くらい詰め込んだもの。その中でも特に詰め込みの多いものをLSIと呼びます。

―イ―

インタフェース＝ＣＰＵと入出力機器など二つの回路や装置をつなぐ場合に、電圧レベルやタイミング、データ形式などの違いを調整するためのハードウエアやソフトウエア。

インタプリタ＝ある言語で書かれたプログラムをコンピュータにわからせるソフトウエアには二通りあります。一つは、そのプログラムをあらかじめ機械語に直してしまい、その機械語プログラムを実行するという方法。（「アセンブラ」と「コンパイラ」の項を参照）もう一つは、機械語に直すことなしに、プログラムの文字を一つ読み込んでは何という文字かを判断して命令を直接実行して行く方法。この第二番目のやり方をするソフトウエアをインタプリタと呼びます。今流行のＢＡＳＩＣはほとんどインタプリタ方式です。

インタラプト＝日本語で「割込み」。あるプログラムの実行を何らかの原因で一時的に中断し、別のプログラムを実行すること。そのプログラムの実行が終わると、元のプログラムにもどる。実際にやることはサブルーチンに似ていますが、どのような形でいつ割込みが起こるかはプログラムを見ただけではわかりません。

―エ―

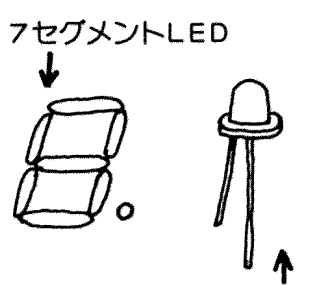
ＬＥＤ＝発光ダイオードのことで、電流が流れると光を出します。簡単な数字や文字が

表示できるように、いくつかのLEDを日の字型に並べたフセグメントLEDというものもあり、電卓やマイコン・キットのごく簡単な出力装置として用いられています。

LSI＝「IC」の項を参照。

―オ―

オーディオ・カセット・インタフェース＝RAMが記憶しているプログラムやデータは電源を切ると消えてしまうので、これを磁気に変えて普通のカセットテープに保存しておくと便利です。そのためのインタフェースを、このように呼びます。プログラムやデータの長期的保存ができ、また規格をそろえておけばある人のカセットの内容を他の人が利用することもできます。この規格の代表的なものが、七五年にアメリカのアマチュア規格となった、カンサス・シティ・スタンダードです。



回路図での記号

―カ―

紙テープ＝幅2・5CMほどの紙のテープに小さい穴をあけて、プログラムやデータを保存しておくためのものです。スピードが遅く取扱いが不便なので、趣味用のマイコンの場合にはオーディオ・カセットの方が

多く用いられています。

カンサス・シティ・スタンダード＝（「オーディオ・カセット・インタフェース」の項を参照。）カンサス・シティで開かれた会合で規格が決定したため、このような名前がつけられました。

―キ―

機械語＝0と1の2進数で表現される言語。コンピュータは基本的には機械語しか理解できません。また、機械語ではプログラムもデータも共に2進数だから、二つを区別するには順番で判断するしかありません。人間が機械語を扱う場合には、2進数ではケタが多くて不便なので、16進数や8進数で表わすのが普通ですが、表わし方が違うだけで意味は同じです。「言語」の項を参照。

キャラクタ＝アルファベット・数字・特殊記号、また時にはカタカナなど、コンピュータが直接扱える記号をキャラクタと呼びます。コンピュータ内部ではキャラクタも2進数に直して扱っているので、キャラクタと2進数とを対応づける規格が必要です。JISコード・ASCIIコードなどがその代表的なものです。

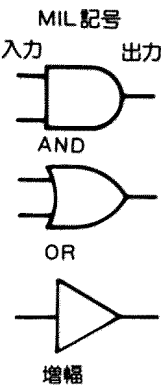
―ク―

クロック＝コンピュータ用語では時計のことではなく、一定の時間間隔をもつパルスのことです。コンピュータのすべての部分は、このクロックに合わせて動作します。高い精度が要求されるので、水晶発振器（略し

てXTAL）を用います。

―ケ―

ゲート＝トランジスタを使った論理回路の基本となるもので、AND・OR・増幅の3種類の回路があります。回路図上でゲートを表わすには、もともとは米軍規格であったMIL記号が一般に用いられています。



（入力または出力端に○印をつけると信号が反転し、NOTの意味になる。）

言語＝人間が行わせたいことをコンピュータに理解させるために使う、キャラクタの集まりを言語と呼びます。コンピュータは基本的には機械語しか理解できないので、ほかの言語の場合には、それをコンピュータに理解させるためのソフトウエアが別に必要です。（「アセンブラ」「インタプリタ」「コンパイラ」の各項を参照）機械語やアセンブリ言語のように、コンピュータの仕組みがわかっていなければ使えない言語を低級言語、BASICなどのように、それがわかっていなくても使える言語を高級言語と呼びます。

K＝「キロ」と読みますが、「1Kバイト」といった場合、これは一〇〇〇バイトのことではありません。二の十乗、つまり一〇二四バイトを指しているのです。

現在、趣味用に使われているほとんどのCPUは、十六ビットでアドレスを表現し、

二の十六乗、つまり六五五三六個の記憶場所を区別できます。この六五五三六のことを六四K（一〇二四の六四倍）という場合と、六五Kという場合があるので、注意して下さい。

―コ―

コンパイラ＝高級言語（「言語」の項を参照）で書かれたプログラムをコンピュータにわからせるための一つの方法として、そのプログラムを機械語に直してしまうという方法があります。そのためのソフトウエアをコンパイラと呼びます。アセンブラと同じく、セルフとクロスの区別があります。同じ高級言語を扱う場合、インタプリタに比べてコンパイラの方がメモリをより多く必要としますが、プログラムを処理する速度は速くなります。

―シ―

シーケンシャル＝「ランダム・アクセス」の項を参照。

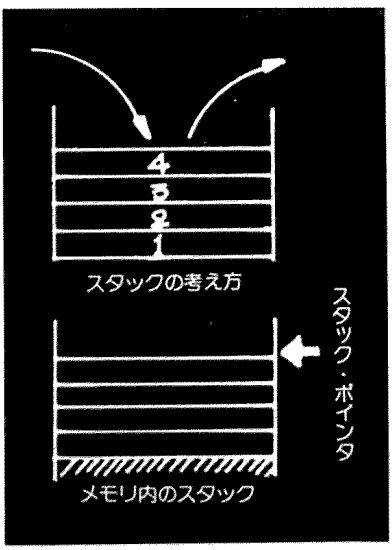
十六進法＝じゅうろくごとに一桁上がる、数の表わし方。ゼロからきゅうまでは算用数字を使えばいいのですが、じゅうからじゅうごまでは一桁で表わす適当な記号がないためアルファベットのA～Fで代用します。例えば十六進法の一〇は十進法の一六にあたります。十六進法を使うと二進法の四桁分が一桁に収まってしまうので、具合がよくなります。

CPU＝セントラル・プロセシング・ユニット（中央処理装置）の略。マイコンと限らず一般のコンピュータで中心機能を持つ部

分です。マイコンの場合は「中央」でない使い方も多いので、CPUでなく、マイクロ・プロセッサとかMPU（マイクロ・プロセシング・ユニット）という人もいます。CPUは制御部、演算部、各種レジスタなどからなり、プログラム・カウンタが示すアドレスの内容（そのアドレスを持つ記憶場所の中味のこと）を読み出してはその命令を解釈実行することで、メモリの中の機械語プログラムを順に処理していきます。コンピュータはCPU、メモリ、入出力装置の三つから構成されているといえます。

――ス――

スタック＝入口と出口が一つになっている棚に物を積むことを考えると、最後に置いた物を必ず最初に取り出すことになります。これがスタックの考え方です。マイコンではメモリの一部分をスタックとして使い、メモリの内どこまでが使われているかを示すためのスタック・ポインタというレジスタをCPU内部にもっています。サブルーチンや、インタラプトの場合のもどり番地の記憶、レジスタの内容の一時的な記憶などに多く用いられます。



―ソ―

ソフトウェア＝コンピュータを動かすのに必要な技術の内、形のないものをソフトウエアと呼びます。具体的にはプログラムの集合を指す言葉です。現在コンピュータと呼ばれているものはすべて、プログラムをあらかじめメモリの中に記憶しておいてそれを順々に実行していくという方式をとっているので、ソフトウエアがなければ何もできません。「ハードウェア」と「ファームウエア」の項を参照。

―タ―

単位＝コンピュータでは、ごく短い時間を表わす言葉がいろいろと出てくるので、それらについてまとめておきます。

1sec	（秒）	＝一秒
1msec	（ミリ秒）	＝千分の一秒
1μsec	（マイクロ秒）	＝百万分の一秒
1nsec	（ナノ秒）	＝十億分の一秒
1psec	（ピコ秒）	＝一兆分の一秒

ダンプ＝メモリやレジスタの内容を、十六進数などの形で出力すること。

―ツ―

通信インタフェース・アダプタ＝データの伝送を一本か二本の信号線を使って行うような入出力機器をつなぐ場合の、インタフェース用ＬＳＩのこと。一語分のデータは一ビットずつ時間をずらして伝送されます。この場合に、一秒間に何ビット伝送できるかを「ボー（Baud）」という単位で表わし

ます。通信インタフェースを必要とする代表的な機器はテレタイプとオーディオカセットで、テレタイプは一一〇ボー、オーディオカセットはカンサス・シティ・スタンダードの場合三〇〇ボーで伝送されます。

―テ―

ディスプレイ＝コンピュータの出力を、後に残らない形で表示する装置。普通は、ブラウン管の上にキャラクタや図形を映し出すものを指します。マイコンの出力機器としてのTVディスプレイについていえば、いろいろな方式がありますが、ビデオRAMと呼ばれる次のような方式が一般的です。つまり、画面を小さい部分に分割したと考え、各々の部分をそれぞれメモリの一バイトに対応させ、CPUは画面に対応しているメモリも普通のメモリも区別せずに扱い、一方、インタフェース回路の方はDMAによって、画面に対応しているメモリの内容を画面上の各部分に表示するという方式です。（「DMA」については、その項を参照）

デジタル回路＝「デジタル」という言葉については「アナログとデジタル」の項でごく簡単に説明しました。デジタル回路では、信号のもつ意味はH（ハイ）かL（ロー）の二通りしかなく信号線の電圧が、ある電圧より高ければH、低ければLと判断されます。このH、Lが代数論理学で使う1、0に対応していてゲートを組合わせることにより、論理式に対応する具体的な回路を作り出すことができます。この意味で、デ

ジタル回路のことを論理回路ということもあります。実際にデジタル回路を構成するICには、ゲートのほかに、フリップフロップやカウンタなどゲートの特別な組合わせを一まとめにしたもの、マルチバイブレータのように抵抗やコンデンサとともに使うものなどがあります。

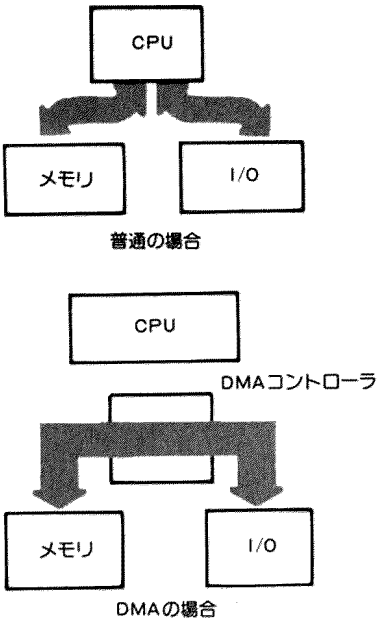
（H（ハイ）、L（ロー）は、電気信号のオン、オフと同じこと。）

デバッグ＝プログラムに間違いがあると、思った通りにコンピュータを動かすことはできません。このプログラムの間違いのことを普通、虫（バッグ）と呼び、プログラムを点検して間違いを直す、つまり虫をとることを、デバッグするといいます。デバッグをしやすくするための特別のハードウエアとソフトウエアをデバッガと呼びます。

テレタイプ＝ＴＴＹとも書きます。もともとは通信用機器。キーボードとプリンタ、紙テープせん孔器、紙テープ読取り器がコンパクトにまとめられていて安価なので、ミニコンや企業用マイコンの入出力機器として多用されています。もっとも、コンパクトかつ安価とはいってもそれは同じような機能をもつ他の製品と比較しての話で、個人が簡単に買って家庭内に持ち込めるようなものではありません。

ＤＭＡ＝ダイレクト・メモリ・アドレスの略。入出力装置などとメモリとの間で、ＣＰＵをはさまず直接にデータを伝送する方式をＤＭＡと呼びます。ＣＰＵでいちいち命令

を実行してデータを送っていたのでは間に合わないような、高速の装置を使う場合に用いられます。



TTL＝トランジスタ・トランジスタ・ロジックの略。バイポーラ型ICには基本回路の違いによっていろいろと種類がありますが、TTLはそれらの内では比較的早く開発されてICの種類も豊富なので、バイポーラ型の内では最も多く用いられています。動作は高速ですが、基本回路が複雑なため小さなパッケージにあまり詰め込むことができず、また消費電力も大きくなります。

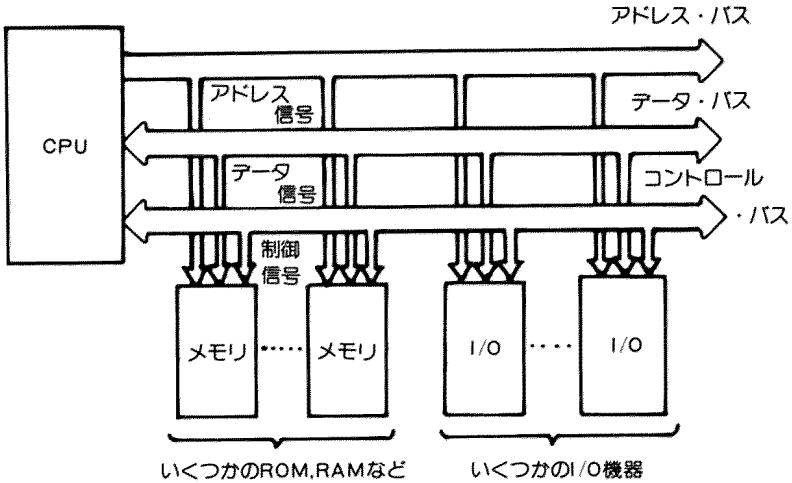
——ニ——

ニーモニック・コード＝機械語命令を覚えやすい文字や記号で表わしたもの。

——ハ——

バイト＝コンピュータの中で扱う情報の単位。一般に、一バイト＝八ビットです。キャラクタを2進数に直すと八ビットで表わすことが多いので、一ワード＝一バイトのマイコンは記号処理に適しています。

バス＝データ、アドレス、制御信号の伝送路



バス

をバスと呼びます。バスの一方の端にCPUがあり、メモリや入出力機器は共通のバスにつながっています。

見かけは、複数本の電線またはプリント基板上のパターンが平行に並んでいるものです。

ハードウェア＝コンピュータを動かすのに必要な技術の内、形のあるものをハードウエアと呼びます。具体的には回路とか装置を指す言葉です。「ソフトウエア」と「ファームウエア」の項を参照。

―ヒ―

ビット＝2進数の一ケタをビットと呼び、コンピュータが表わせる情報の最小単位です。

ビット・スライス＝普通のCPUでは一ワー

ドの長さは決まっていて変えられませんが、ビット・スライス型マイクロ・プロセッサではaビットのマイクロ・プロセッサを並列にb個並べることによって、一ワード＝a×bビットのマイコンを作ることができます。

―フ―

ファームウェア＝ROMに固定されているプログラムについては、回路がプログラムを覚えていると考えられるわけで、これをハードウェアとソフトウェアの中間という意味で、ファームウェアと呼ぶことがあります。

フラグ＝「旗」のことで、CPUが今、どのような状態にあるかとか演算結果がどうなったかなどを表わすためのビット。コンディション・コードともいい、CPU内にいくつかあるフラグの全体を、コンディション・コード・レジスタまたはステータス・ワードなどと呼ぶ。フラグの例としては、次のようなものがあります。ほかにも、マイクロプロセッサによっていろいろあります。

インタラプト・フラグ（CPUが割込み可能ならばオン）

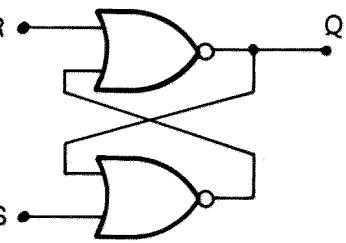
ゼロ・フラグ（演算結果がゼロならばオン）

キャリー・フラグ（演算の結果、最上位桁からくり上がりがあればオン）

フリップフロップ＝0と1という二つの安定状態をもち、入力信号によってどちらの安定状態をとるかが決まる回路をフリップフロップと呼びます。最も簡単な例は図のよ

うな回路で、一入力RとS、一出力Qをもつ。今、S(セット)を1、R(リセット)を0にするとQは1となり、Sを0に戻しても変わりません。次にRを1にするとQは0となり、やはりRを0に戻しても変わりません。つまり、SとRのいずれを1にするかによって、0と1のいずれでQが安定するかが決まるのです。ただしこの例では、SとRを同時に1にすると何が起こるかわかりません。この例は最も簡単なRS-FF(フリップフロップ)というタイプで、ほかにはJK-FF、D-FF、T-FFがあり、カウンタやスタティックRAMなどを構成する基本となっています。

フローチャート=プログラムの論理的・時間的流れが一目でわかるように、いろいろな形のワクと矢印で表わした図。



S	R	Q
0	0	不変
1	0	1
0	1	0
1	1	?

フリップフロップの例

フロッピィ・ディスク=ランダム・アクセスのできる補助記憶装置。磁性粉の塗られたプラスチックの薄い円盤が袋に入っています。軽くて扱いやすく安価なので、多用されています。ディスケット、フレキシブル・ディスクとも呼ばれます。一回り小さな

ミニ・フロッピィというのもあり、フロッピィが直径一〇センチなのに対して、直径十二センチです。

——へ——

並列処理＝一ワードの各ビットを2個以上（普通は一ワードのビット数に等しい）の回路で同時に処理すること。現在のマイコンでは並列処理が一般的です。

BASIC＝「ベイシック」と読みます。ビギナーズ・オールパーパス・シンボリック・インストラクション・コード（初心者向き汎用記号命令体系）の略。高級言語の一つで、FORTRANを簡単にした会話型言語として開発されたものですが、現在のところ、マイコン用高級言語の主流となっています。方言がいろいろあり、あるインタプリタで使えたからといって、同じプログラムが別のインタプリタでも使えるとは限りません。規格を作ろうという動きもあります。

——ホ——

ボー＝「通信インタフェース・アダプタ」の項を参照。

補数＝二進数の場合、2の補数と1の補数の2種類があります。

2の補数とは、各桁の0・1を反対にして1を加えたもので負の数を表現する方法で、最上位桁が0の場合には正、1の場合には負とみなします。これを使うと減算を加算におきかえることができます。1の補数と

2の補数の考え方

0101　（10進法だと＋5）
↓
これの2の補数を作るには、まず0・1を反対にして（これは1の補数）
1010
↓
そして1を加える
1011　（これが＋11ではなく－5を表わしていると考えるのが2の補数の考え方）

例えば

7－5＝2

（0111－0101＝0010）

を計算するのに

7＋(－5)

と考えて、2の補数を使うと

```
  0111
+)1011
-----
 10010
```

となり、最上位桁のあふれを無視すると、確かに0010となる。

は、各桁の0・1を反対にしたもので、論理学でのNOTに対応しています。

ポート＝CPUと入出力機器とのインタフェースで、入出力機器にタイミングを合わせるためにデータを一時的に記憶するレジスタを、特にポートと呼ぶことがあります。この制御のために特別に作られたLSIを使うのが普通です。

―メ―

メモリ＝広い意味では、演算や制御に必要な

プログラム・データを貯えたり、得られた結果をしまっておくためのハードウエアをメモリ（記憶装置）と呼びます。記憶装置は主記憶装置と補助記憶装置の二つに大きく分けられます。主記憶装置は、記憶容量はあまり大きくなくてもいいのですが、アクセス・タイムが短くかつランダム・アクセスが可能でなければならず、マイコンの場合、ＩＣメモリが使われます。普通はメモリというと、この主記憶装置を指します。これに対して、補助記憶装置としては、磁気テープ、フロッピィ・ディスクなど、記憶容量の大きなものが用いられます。

ＩＣメモリには、書き込みが自由にできるものとできないものがあって、前者をＲＡＭ（ランダム・アクセス・メモリ）、後者をＲＯＭ（リード・オンリー・メモリ）と呼びますが、正確にいうとＲＯＭもランダム・アクセス・メモリなので、書込み自由なメモリのことをＲＷＭ（リード・ライト・メモリ）と呼ぶ人もいます。電源を切ると、ＲＡＭの内容は消えてしまうが、ＲＯＭの内容は消えません。つまり、ＲＯＭにプログラムを入れておけば、電源を入れるたびごとにあらためて書きこむという必要はなくなります。

なお、オーディオ・カセットや紙テープは記憶装置ではありません。入出力インタフエースを通してＣＰＵとつながっているからです。

――モ――

モニタ―コンピュータはソフトウエアがなければ何もできません。人間が何かをやらせ

たい場合、プログラムをＲＡＭに書き込んで、実行開始番地をＣＰＵに教えてやらないと、コンピュータは動作しないのです。書込みなどはハードウエアでやってもいいのですが、このようなことをするプログラムをあらかじめＲＯＭに入れておいて、電源が入ったら直ちにこのＲＯＭ内のプログラムが実行されるようにしておけば、便利になります。このようなソフトウエアをモニタと呼び、他には、入出力機器を制御するプログラムや簡単なデバッガなども含まれています。

ＭＯＳ＝「モス」と読みます。メタル・オキサイド・セミコンタクタの略。作り方で分類した場合のトランジスタの一種で、また、このトランジスタを基本として作られたＩＣもＭＯＳと呼びます。ＴＴＬに代表されるバイポーラ型ＩＣに比べて、ＭＯＳ型ＩＣは構造が簡単なので小さい面積に多くをつめ込むことができ、また消費電力も小さいのですが、信号処理のスピードが遅いという欠点があります。

―ラ―

ランダム・アクセス＝記憶装置に対して読出し、書込みをすることを、アクセスするといいますが、記憶場所のアドレスによらずアクセス・タイムがほぼ一定であることを、ランダム・アクセスと呼びます。ランダム・アクセスのできる記憶装置としては、ＩＣメモリやフロッピィ・ディスクなどがあります。ランダム・アクセスの反対語をシーケンシャルといい、磁気テープなどがシ

ーケンシャルな記憶装置の代表です。

―リ―

リフレッシュ＝ダイナミックＲＡＭでは放っておくと記憶した内容が失われてしまうので、一定時間ごとに信号パルスを与えてやらねばなりません。このことをリフレッシュするといいます。

―レ―

レジスタ＝何ビットかのデータを一時的に記憶するための回路。命令実行や演算、データ転送を容易にするために用いられます。主なレジスタとしては次のようなものがあります。

アキュムレータ
汎用レジスタ
インデックス・レジスタ
プログラム・カウンタ
スタック・ポインタ

―ワ―

ワード＝日本語で「語」。コンピュータでひとまとまりとして処理される情報の単位。ＣＰＵの種類によって一ワードの長さは違い、マイコンの場合、四ビット、八ビット、十二ビット、十六ビットなどがあります。
（「ビットスライス」の項参照）

〈著者紹介〉

山科敦之（やましなあつし）

一九五〇年　北九州市に生れる
東京大学仏文科に進み、コンピュータの講義に触発され、情報科学を専攻する
現在東京大学新聞研究所研究生
現住所　中野区鷺宮3―8―2

吉田紀彦（よしだのりひこ）

一九五四年　東京に生れる
東京大学工学部計数学科に進む
一九七八年の東大五月祭に「ヘビとネズミ」のマイコン・プログラムを考案、発表
現住所　府中市本町2―25―3

よくわかるマイコン入門

1979年6月25日　発行　©

著者　山科敦之
　　吉田紀彦
発行者　富永弘一
印刷所　サンアート印刷

発行所　東京都台東区台東2丁目24　株式会社新星出版社
電話（831）0743　郵便番号110　振替東京 4-72233

●よくわかる　ＢＣＬ入門　★大河原真著
●よくわかる　オーディオ入門　★森　耕一著
●よくわかる　マイコン入門　★山科敦之著
●やさしい　ラジオ工作入門　★黒岩文雄著

趣味の技術入門

●よくわかる

マイコン入門

新星出版社
¥680
2055 35141 3151